INTRODUCTION TO THE Z80 MICROCOMPUTER

ADI J. KHAMBATA

INTRODUCTION TO THE Z80 MICROCOMPUTER

ADI J. KHAMBATA
St. Paul Technical-Vocational Institute

JOHN WILEY & SONS
New York Chichester Brisbane Toronto Singapore

Copyright © 1982 by John Wiley & Sons, Inc.

All rights reserved.

Reproduction or translation of any part of
this work beyond that permitted by Section
107 or 108 of the 1976 United States Copyright
Act without the permission of the copyright
owner is unlawful. Requests for permission
or further information should be addressed to
the Permissions Department, John Wiley & Sons, Inc.

ISBN 0-471-86167-7
Printed in the United States of America

10 9 8 7 6 5 4 3 2 1

To my brother Navroze (Pinky)

PREFACE

This book is intended as a supplement to the textbook *MICROPROCESSORS/MICROCOMPUTERS: Architecture, Software, and Systems* (written by this author and published by JOHN WILEY & SONS, INC., New York, January 1982). It may also be used as a supplement for other basic texts, or as a brief stand-alone introduction to the Z80.

During the planning and writing of *MICROPROCESSORS/MICROCOMPUTERS*, we encountered the following problem. In addition to offering theoretical descriptions and discussions, we wanted to describe at least one real, existing microcomputer system. However, including a description of any one particular product in that text would immediately have dated the publication and also made it unacceptable to other users who might be committed to another product. This dilemma was resolved by adopting a unique approach. A series of separate, softcover supplements, each covering a popular microcomputer system, will accompany the basic textbook. Four such supplements are presently planned. They will cover the ZILOG Z80, the INTEL 8080/8085, the MOTOROLA MC6800, and the 6502 by MOS TECHNOLOGY. This approach will enable us to introduce additional supplements on future products that find acceptance in educational institutions as well as update existing supplements as needed. It also allows users of the main textbook to purchase only the supplements that are of direct use for their specific needs. In addition to the paperback supplements, separate laboratory manuals—on the most popular microcomputers—will also be published.

This publication is the first product supplement. It is based primarily on the Z80 courses that I have designed and taught at St. Paul Technical–Vocational Institute, St. Paul, Minnesota, and in local industries over the past several years. As a result of this experience, I have introduced a number of features in this book. First, each chapter starts with a brief listing of chapter objectives, in order to give students a quick overview of the specific items that will be covered in each chapter. Upon completing a study of the chapter students will be able to review the objectives, thereby (hopefully) gaining further understanding of the material covered in that chapter.

Second, since this book may be used as a companion to my hardcover text, references to that book are included for the principal topics covered in each chapter. The references are made by both chapter and sections within the chapters. I feel that this simple convenience will save students a lot of time and effort when they wish to refer to the main textbook for a quick review of any particular topic.

Third, at the end of each chapter, numerous review questions are included. To aid both students and instructors, the questions are structured in formats such as true/false, multiple choice, and fill-in-the-blanks. For most questions, several possible answers are provided in parentheses at the end of the question. The student has only to select the appropriate answer. However, selection of the correct answer requires an understanding of the subject matter of the chapter. Furthermore, the earlier questions in any chapter are simple, but they become progressively more complex and more challenging.

Finally, two distinct styles of presentation are introduced in each chapter, namely, a list style and a descriptive prose style. The descriptive prose style is certainly easier to read and, in many cases, easier to understand since more explanation is provided. However, classroom experience has indicated to me that the prose style does very little to aid students in remembering specific technical points or events. For instance, if a certain operation involves six tasks to be performed in a specified sequence, it is much easier for a student to remember an enumeration of those tasks in six discrete steps than to remember them from a prose description.

The Z80 CPU chip and its principal support chips are described in this book. In addition, chapters are included for initializing, or programming, three of the most commonly used support chips (the PIO, the CTC and the DMA chips). The hardware architecture of the Z80/CPU chip is described in Chapter 1. This includes the principal registers and the various addressing modes as well as package pin assignments, presented in convenient tabular form. Six groups of Z80 instructions are presented and described in Chapter 2. They include LOAD instructions (8 and 16 bits), STACK PUSH/POP, EXCHANGE, BLOCK TRANSFER/SEARCH, 8-bit arithmetic/logic and 16-bit arithmetic instructions. This chapter also includes detailed explanations (aided by tables) of instruction formats and various symbols used in the instructions. Chapter 3 continues with the instruction set and covers ROTATE/SHIFT, BIT MANIPULATION, JUMP and CALL/RETURN instruction groups. Chapter 4 completes the instruction set. It includes the INPUT/OUTPUT instructions involved in the interrupt system of the Z80. Chapter 5 concludes the discussion of the CPU chip. Timing charts of the various functions (both with and without WAITstates) are included in this chapter.

Chapter 6 covers the Parallel Input Output (PIO) chip. It includes the architecture of the system, the interrupt control protocol and the interrupt timing involved in the three operational modes. Chapter 7 is a continuation of the previous chapter. It introduces the concept of customizing the PIO chip for specific applications by initializing it accordingly. Two specific examples of different applications are included. The Counter/Timer Circuit (CTC) chip is described in Chapter 8. Both the counter and timing functions are covered along with the various operating modes. Chapter 9 shows how the CTC can be preprogrammed for specific applications. Examples for both counter and timer applications are included.

The Direct Memory Access (DMA) chip is covered in Chapter 10. Operations involving transfer only, search only, and combined search-transfer are described. Also, the byte-at-a-time, burst, continuous and transparent modes of operation are covered in reasonable detail. Programming the DMA chip for various applications is covered in Chapter 11 with several examples. The Serial Input Output (SIO) chip is the subject of Chapter 12. Both the byte-oriented and the bit-oriented protocols are included. Both asynchronous and synchronous modes of operation are covered. The operation and use of the Cyclic Redundancy Check (CRC) character is also described.

I would like to thank the many people who helped me during the writing and publication of this book. At Wiley, Judy Green, Engineering Technology Editor, Susan Weiss, Associate Editor, and Alice Tufel, Senior Administrative Assistant, were extremely helpful during the entire writing and publication process. Their efforts and cooperation are most appreciated. Dr. George Richter, Technical Division Manager at St. Paul Technical-Vocational Institute, constantly encouraged and supported me. Several of my students (past and present) and former engineering colleagues at Sperry Univac reviewed parts of the manuscript and offered many valuable comments and suggestions; special contributions were made by Kenneth Jarosch, Philip Gaines, Walter Knights, and Jay

Meyers. I would also like to thank the following reviewers whose comments and suggestions were invaluable in preparing the final version of the manuscript: Lou Gross, Columbus Technical Institute; Irving L. Kosow, Southern Technical Institute; J.W. Carter, Memphis State University; David Hata, Portland Community College; Peter Julian, Columbus Technical Institute; Arthur H. Seidman, Pratt Institute; David Terrell, I.T.T. Technical Institute; Charles Van Buren, DeVry Technical Institute.

As before, this book is a Khambata family project. My daughter Pixie, assisted by my wife Ruth, typed the original manuscript; corrections and proofreading were handled by my son Jim and his wife Shelly; and all diagrams and timing charts were drawn by my son Danny and his wife Renee. I thank all of them.

Adi J. Khambata

St. Paul, Minnesota
February 7, 1982

ACKNOWLEDGMENTS

The author gratefully acknowledges and thanks ZILOG, INC., Cupertino, California, for granting permission to use and reproduce in this book block diagrams and timing charts, as well as the instruction set, of the Z80 Microcomputer from ZILOG published Technical Manuals. These manuals and the material used from them are identified in the following list.

Since this product supplement is primarily intended for student use in the classroom, minor modifications and additions are made in the block diagrams and timing charts to conform to explanatory material in the text and to aid the student in learning and understanding the subject matter.

ZILOG and Z80 (whichever applicable) are/is a trademark(s) of Zilog, Inc., with whom the author or publisher is not connected.

Reproduced by permission © 1977 & 1981 by Zilog, Inc. This material shall not be reproduced without written consent of Zilog, Inc.

From the Z80-CPU & Z80A-CPU TECHNICAL MANUAL © 1977

Page 4	Fig. 2.0-2	Z80 CPU Register Configuration
Page 3	Fig. 2.0-1	Z80 CPU Block Diagram
Page 39		Flag Register Format
Page 41	Table 6.0-1	Summary of Flag Operation (Note: Some of the notation symbols have been changed to accommodate the types available on my typewriter.)
Pages 43-54	Tables 7.0-1 thru 7.0-11	Summary of OP codes and execution times (Note: Flag symbols have been changed to accommodate available types on my typewriter. The columns for M cycles and number of T cycles are not included. Finally word descriptions of each word are added.)
Page 11	Fig. 4.0-0	Basic CPU Timing example.
Page 12	Fig. 4.0-1	Instruction OP code Fetch
Page 13	Fig. 4.0-1A	Instruction OP code Fetch with WAIT states
Page 13	Fig. 4.0-2	Memory Read or Write Cycles
Page 14	Fig. 4.0-2A	Memory Read or Write Cycles with WAIT states
Page 15	Fig. 4.0-3	Input or Output Cycles
Page 15	Fig. 4.0-3A	Input or Output Cycles with WAIT states
Page 16	Fig. 4.0-4	Bus Request/Acknowledge Cycle
Page 16	Fig. 4.0-5	Interrupt Request/Acknowledge Cycle
Page 18	Fig. 4.0-6	Non-Maskable Interrupt Request/Operation
Page 18	Fig. 4.0-7	Halt Exit

From the Z80-PIO & Z80A-PIO TECHNICAL MANUAL © 1977

Page 3	Fig. 2.0-1	PIO Block Diagram
Page 3	Fig. 2.0-2	Port I/O Block Diagram
Page 13	Fig. 5.0-1	Mode O (Output) Timing
Page 13	Fig. 5.0-2	Mode 1 (Input) Timing
Page 14	Fig. 5.0-3	Port A, Mode 2 (Bidirectional) Timing
Page 14	Fig. 5.0-4	Mode 3, Bit Control Mode Timing
Page 15	Fig. 6.0-1	Interrupt Acknowledge Timing
Page 17	Fig. 7.0-1	A method of extending the interrupt priority daisy chain

From the Z80-CTC & Z80A-CTC TECHNICAL MANUAL © 1977

Page 2	Fig. 2.0-1	CTC Block Diagram
Page 3	Fig. 2.0-2	Channel Block Diagram
Page 16		CTC Write Cycle
Page 17		CTC Read Cycle
Page 18		CTC Counting and Timing Diagram
Page 19		Interrupt Acknowledge Timing Diagram
Page 20		Return from Interrupt Timing Diagram

From the Z80-SIO TECHNICAL MANUAL © 1977

Page 1		Z80-SIO Block Diagram
Page 6	Fig. 4	Transmit and Receive Data Path
Page 13	Fig. 7	Synchronous Formats
Page 21	Fig. 8	Transmit/Receive SDLC/HDLC Message Format
Page 41	Fig. 14d	Return from Interrupt Cycle
Page 41	Fig. 14c	Interrupt Acknowledge Cycle
Page 41	Fig. 14b	Write Cycle
Page 41	Fig. 14a	Read Cycle

From the Z80-DMA TECHNICAL MANUAL (January 1981) © 1981

Page 2-3	Fig. 5	Basic Functions of Z80 DMA
Page 8-7	Fig. 62	Variable-Cycle and Edge Timing
Page 8-7	Fig. 63	WAIT line Sampling in Variable-Cycle Timing
Page 8-1	Fig. 48	CPU-to-DMA Write Cycle Requirements
Page 8-1	Fig. 49	CPU-to-DMA Read Cycle Requirements
Page 8-2	Fig. 50	Sequential Memory-to-I-O Transfer Standard Timing
Page 8-2	Fig. 51	Sequential I/O-to-Memory Transfer Standard Timing
Page 8-4	Fig. 54	Bus Request and Acceptance Timing
Page 8-4	Fig. 56	Bus Release on End-of-Block (Burst & Continuous Modes)
Page 8-4	Fig. 58	Bus Release on Not-Ready (Burst Mode)
Page 8-4	Fig. 57	Bus Release on Match (Burst and Continuous Modes)
Page 8-4	Fig. 55	Bus Release in Byte Mode

CONTENTS

1 THE Z-80 CPU ARCHITECTURE

CHAPTER OBJECTIVES *1*

TEXTBOOK REFERENCES *1*

1-1 INTRODUCTION *2*

1-2 THE Z-80 SYSTEM *2*

1-2.1 Principal Hardware Components *2*

1-2.2 The Development System *3*

1-3 PRINCIPAL FEATURES OF THE Z-80 CPU *3*

1-4 THE CPU ARCHITECTURE *5*

1-4.1.1 *GP Registers, Accumulators and Status Flag Registers* *5*

1-4.1.2 *Special-Purpose Registers* *6*

1-4.2 The Arithmetic Logic Unit (ALU) *9*

1-4.3 Instruction Register, Decode and Control *10*

1-4.4 Program Status Word (F and F') *10*

1-5 ADDRESSING MODES *13*

1-5.1 Implied Addressing *13*

1-5.2 Register Addressing *13*

1-5.3 Register Indirect Addressing *13*

1-5.4 Bit Addressing *13*

1-5.5 Immediate Addressing *14*

1-5.6 Immediate Extended Addressing *14*

1-5.7 Extended Addressing (Direct Addressing Mode) *14*

1-5.8 Indexed Addressing *14*

1-5.9 Relative Addressing *14*

1-5.10 Modified Page Zero Addressing *15*

1-6 PACKAGE PIN ASSIGNMENTS AND FUNCTIONS *15*

1-7 REVIEW QUESTIONS *21*

2 THE Z-80 INSTRUCTION SET (PART ONE)

CHAPTER OBJECTIVES *29*

TEXTBOOK REFERENCES *29*

2-1 INTRODUCTION *30*

2-2 CATEGORIES OF INSTRUCTIONS *30*

2-2.1 The Load and Exchange Group *30*
2-2.2 The Block Transfer and Search Group *30*
2-2.3 The Arithmetic and Logic Group *31*
2-2.4 The Rotate and Shift Group *31*
2-2.5 The Bit Manipulation Group (Set, Reset and Test) *31*
2-2.6 The JUMP, CALL and RETURN Group *31*
2-2.7 The Input/Output Group *31*
2-2.8 The Basic CPU Instructions Group *32*
2-3 INSTRUCTION FORMAT AND SYMBOLS *32*
2-3.1 The Generalized Instruction Format *32*
2-3.2 The OP Code *32*
2-3.3 The Mnemonic Structure *34*
2-3.4 Symbolic Expressions of Operations and Functions *35*
2-3.5 The Status Register Symbols *39*
2-3.6 The Register Pair Codes *39*
2-4 THE INSTRUCTION SET *42*
2-5 REVIEW QUESTIONS *60*

3 THE Z-80 INSTRUCTION SET (PART TWO)
CHAPTER OBJECTIVES *65*
TEXTBOOK REFERENCES *65*
3-1 INTRODUCTION *65*
3-2 SHIFT OPERATIONS *66*
3-2.1 Arithmetic Shifts *66*
3-2.1.1 *Shift Left Arithmetic (SLA)* *66*
3-2.1.2 *Shift Right Arithmetic (SRA)* *67*
3-2.1.3 *Shift Right Logic (SRL)* *67*
3-2.2 Register/Memory Symbols for Rotate/Shift *68*
3-2.3 The Mnemonic Convention *68*
3-2.4 OP Code Construction *70*
3-3 ROTATE AND SHIFT GROUP *72*
3-4 BIT MANIPULATION GROUP *75*
3-5 THE JUMP INSTRUCTIONS GROUP *77*
3-6 THE CALL AND RETURN GROUP *80*
3-7 REVIEW QUESTIONS *83*

4 THE Z-80 INSTRUCTION SET & INTERRUPT RESPONSE
CHAPTER OBJECTIVES *87*
TEXTBOOK REFERENCES *87*
4-1 INTRODUCTION *88*
4-2 THE I/O INSTRUCTION GROUP *89*

4-3 THE BASIC CPU INSTRUCTION GROUP *94*
4-4 THE INTERRUPT SYSTEM *96*
4-4.1 The Objectives *96*
4-4.2 The Non-Maskable Interrupt (NMI) *96*
4-4.3 The Maskable Interrupt (INT) *96*
4-4.4 The INT Enable-Disable Flip-Flop *97*
4-4.5 CPU Response to Non-Maskable Interrupts (NMI) *98*
4-4.6 CPU Response to Maskable Interrupts (INT) *98*
4-4.6.1 *Mode O* *98*
4-4.6.2 *Mode 1* *98*
4-4.6.3 *Mode 2* *99*
4-5 REVIEW QUESTIONS *99*

5 THE Z-80/CPU TIMING AND CONTROL
CHAPTER OBJECTIVES *105*
TEXTBOOK REFERENCES *105*
5-1 THE FUNDAMENTAL INSTRUCTION CYCLE *106*
5-2 THE OP CODE FETCH CYCLE *107*
5-2.1 Without WAIT States *107*
5-2.2 With WAIT States *109*
5-3 MEMORY READ AND WRITE CYCLES *111*
5-3.1 Without WAIT States *111*
5-3.2 With WAIT States *113*
5-4 I/O READ AND WRITE CYCLES *113*
5-4.1 Without WAIT States *113*
5-4.2 With WAIT States *116*
5-5 BUS REQUEST/ACKNOWLEDGE CYCLES *116*
5-6 MASKABLE INTERRUPT REQUEST/ACKNOWLEDGE CYCLE *119*
5-7 NON-MASKABLE INTERRUPT REQUEST/ACKNOWLEDGE CYCLE *119*
5-8 EXIT FROM HALT INSTRUCTION CYCLE *122*
5-9 REVIEW QUESTIONS *122*

6 Z-80 PARALLEL INPUT OUTPUT CHIP
CHAPTER OBJECTIVES *129*
TEXTBOOK REFERENCES *129*
6-1 INTRODUCTION *130*
6-2 PRINCIPLE FEATURES OF THE PIO *130*
6-3 THE PIO ARCHITECTURE *130*
6-3.1 The Generalized PIO Block Diagram *130*
6-3.2 The Port Logic Block Diagram *131*
6-3.3 The Interrupt Control Protocol *134*

6-4 PACKAGE PIN ASSIGNMENTS AND FUNCTIONS *134*
6-5 THE PIO INTERRUPT TIMING *143*
6-5.1 Mode O - Output Byte Mode *143*
6-5.2 Mode 1 - Input Byte Mode *145*
6-5.3 Mode 2 - Bidirectional Mode *145*
6-5.4 Mode 3 - Bit Control Mode *148*
6-5.5 Interrupt Acknowledge and Servicing *150*
6-5.6 Extending the Daisy Chain *150*
6-6 REVIEW QUESTIONS *154*

7 PROGRAMMING THE PIO CHIP
CHAPTER OBJECTIVES *159*
TEXTBOOK REFERENCES *159*
7-1 INTRODUCTION *160*
7-2 THE PROGRAMMING WORDS *160*
7-2.1 Word for Mode Set *160*
7-2.2 The Interrupt Vector *161*
7-2.3 Set Interrupt Control Byte *161*
7-2.4 Interrupt Enable Flip-Flop Set *162*
7-3 PROGRAMMING THE PIO FOR A CONTROL APPLICATION *163*
7-4 PROGRAMMING THE PIO FOR A BYTE TRANSFER APPLICATION *167*
7-5 REVIEW QUESTIONS *170*

8 Z-80 COUNTER/TIMER CIRCUIT CHIP
CHAPTER OBJECTIVES *175*
TEXTBOOK REFERENCES *175*
8-1 INTRODUCTION *175*
8-2 PRINCIPAL FEATURES OF THE CTC *176*
8-3 THE CTC ARCHITECTURE *176*
8-3.1 Generalized CTC Block Diagram *176*
8-3.2 The Channel Logic Block Diagram *178*
8-3.3 Interrupt Control Logic and Protocol *181*
8-4 PACKAGE PIN ASSIGNMENTS AND FUNCTIONS *182*
8-5 THE OPERATING MODES *190*
8-5.1 Setting up the CTC Chip *190*
8-5.2 Operating in the Counter Mode *190*
8-5.3 Operating in the Timer Mode *191*
8-6 THE CTC TIMING *192*
8-6.1 The CTC Write Cycle *192*
8-6.2 The CTC Read Cycle *194*

8-6.3 The CTC Counter Mode *194*
8-6.4 The CTC Timer Mode *196*
8-6.5 Interrupt Acknowledge Cycle Timing *197*
8-6.6 Return from Interrupt Timing *199*
8-7 REVIEW QUESTIONS *199*

9 PROGRAMMING THE CTC CHIP
CHAPTER OBJECTIVES *205*
TEXTBOOK REFERENCES *205*
9-1 INTRODUCTION *205*
9-2 THE PROGRAMMING WORDS *206*
9-2.1 The Interrupt Vector *206*
9-2.2 The Channel Control Word *208*
9-2.3 The Time Constant Word *209*
9-3 PROGRAMMING FOR COUNTER APPLICATIONS *210*
9-4 PROGRAMMING FOR TIMER APPLICATIONS *213*
9-5 REVIEW QUESTIONS *216*

10 Z-80 DIRECT MEMORY ACCESS CHIP
CHAPTER OBJECTIVES *221*
TEXTBOOK REFERENCES *222*
10-1 INTRODUCTION *222*
10-2 PRINCIPAL FEATURES OF THE DMA *223*
10-2.1 General and Electrical Features *223*
10-2.2 Operational Features *223*
10-3 THE DMA ARCHITECTURE *224*
10-4 PACKAGE PIN ASSIGNMENTS AND FUNCTIONS *227*
10-5 FUNCTIONAL OPERATION OF THE DMA *235*
10-5.1 The Classes of Operations *235*
10-5.2 Modes of Operation *236*
10-5.3 The Interrupt System *237*
10-5.4 Pulse Generation *238*
10-5.5 Command and Control Bytes *238*
10-5.6 Reading from Internal Registers *238*
10-5.7 Variable Cycle Length Time *239*
10-5.8 Port Addressing *239*
10-5.9 Auto Restart *239*
10-6 THE DMA TIMING *240*
10-6.1 The Variable Cycle Timing *240*
10-6.2 DMA Inactive State Timing *240*

10-6.2.1 *Command/Control Byte Write Cycle* *240*
10-6.2.2 *DMA Register Read Cycle* *240*
10-6.3 DMA Active State Timing *242*
10-6.3.1 *Memory-to-I/O Transfers* *242*
10-6.3.2 *Memory-to-Memory Transfers* *243*
10-6.3.3 *I/O-to-Memory Transfers* *243*
10-6.3.4 *I/O-to-I/O Transfers* *247*
10-6.4 Bus Request and Acceptance Timing *247*
10-6.5 Bus Release Timing for Burst and Continuous Modes *247*
10-6.5.1 *At End of Block* *247*
10-6.5.2 *On "Not Ready"* *247*
10-6.5.3 *On "Match"* *250*
10-6.6 Bus Release Timing for Byte-at-a-Time Mode *250*
10-7 REVIEW QUESTIONS *252*

11 PROGRAMMING THE DMA CHIP
CHAPTER OBJECTIVES *259*
TEXTBOOK REFERENCES *259*
11-1 INTRODUCTION *260*
11-2 COMMAND WORDS FOR SET-UP *260*
11-2.1 The WRO Group *261*
11-2.2 The WR1 Group *261*
11-2.3 The WR2 Group *264*
11-2.4 The WR3 Group *265*
11-2.5 The WR4 Group *265*
11-2.6 The WR5 Group *269*
11-2.7 The WR6 Group *270*
11-2.8 The Status Register *272*
11-3 PROGRAMMING THE DMA FOR TRANSFER ONLY OPERATIONS *272*
11-3.1 I/O-to-Memory Transfer *273*
11-3.2 Memory-to-I/O Transfer *275*
11-4 PROGRAMMING THE DMA FOR SEARCH OPERATIONS *277*
11-5 PROGRAMMING THE DMA FOR SEARCH-TRANSFER OPERATIONS *279*
11-6 REVIEW QUESTIONS *281*

12 Z-80 SERIAL INPUT OUTPUT CHIP
CHAPTER OBJECTIVES *285*
TEXTBOOK REFERENCES *285*
12-1 INTRODUCTION *286*
12-2 PRINCIPAL FEATURES OF THE SIO *286*

12-3 THE SIO ARCHITECTURE *287*
12-3.1 The Generalized SIO Block Diagram *287*
12-3.2 The Receive Data Path *289*
12-3.2.1 *The Asynchronous Mode* *289*
12-3.2.2 *The Synchronous Mode* *292*
A. BYTE-ORIENTED PROTOCOLS *292*
B. BIT-ORIENTED PROTOCOL (SDLC MODE) *294*
12-3.3 The Transmit Data Path *294*
12-3.3.1 *The Asynchronous Mode* *296*
12-3.3.2 *The Synchronous Mode* *296*
12-4 PACKAGE PIN ASSIGNMENTS AND FUNCTIONS *297*
12-4.1 The CPU Side Pins *297*
12-4.2 The I/O Side Pins *302*
12-5 The SIO TIMING *308*
12-5.1 The Read Cycle *308*
12-5.2 The Write Cycle *308*
12-5.3 The Interrupt Acknowledge Cycle *311*
12-5.4 The Return from Interrupt Cycle *311*
12-6 NOTES ON CRC OPERATION *311*
12-6.1 Introduction *311*
12-6.2 Principle of Operation *314*
12-6.3 The CRC Generator in the Transmitter *315*
12-6.4 The CRC Generator in the Receiver *318*
12-6.5 The Comparison Process in the Receiver *318*
12-7 REVIEW QUESTIONS *320*

ANSWERS TO SELECTED REVIEW QUESTIONS *326*
INDEX *334*

1 THE Z-80 CPU ARCHITECTURE

CHAPTER OBJECTIVES

The objectives of this chapter are as follows:

1. To introduce students to the Z-80 μC system and its principal hardware components.
2. To describe the hardware architecture of the Z-80/CPU.
3. To describe the general-purpose (GP) and the special-purpose (SP) registers and their respective functions.
4. To present the functions performed by the arithmetic-logic-unit (ALU).
5. To discuss the program status word (PSW) and present its format.
6. To describe and discuss the 12 addressing modes used in the Z-80/CPU.
7. To discuss the functions of the CPU package pins and present them in convenient tabular form.

TEXTBOOK REFERENCES

For the convenience of students who wish to review the relevant material of this chapter in the main (hardcover) textbook, the following sections and/or pertinent chapters are suggested:

1. For general review of CPU registers and counters - Sec. 1-3.2
2. For Index Register - Secs. 5-6 and 5-7
3. For Program Status Word - Sec. 3-5.3
4. For Addressing Modes - Chapter 5

5. For Interrupt Page Address Register - Secs. 9-5 and 9-7

1-1 INTRODUCTION

The Z-80 microcomputer system is designed and manufactured by ZILOG, INC. of Cupertino, California. At the present time, MOSTEK, INC. of Carrollton, Texas is the official second source for the Z-80 products.

The Z-80 devices are designed by personnel who were formerly employed by INTEL CORPORATION, and had designed the INTEL 8080A which is an enhanced version of the INTEL 8008 μC. Thus the Z-80 naturally tends to have many similarities with the 8080A and can be considered as an enhancement of the 8080A. INTEL's enhanced version of the 8080A is the 8085A.

1-2 THE Z-80 SYSTEM

1-2.1 Principal Hardware Components

The Z-80 μC system consists of 5 principal hardware components which are briefly described below:

1. The Z-80/CPU - This is the central processing unit which contains the arithmetic logic unit (ALU) and the other logical components required for performing the usual CPU functions.

2. The Z-80/PIO - This is a programmable, 2-port, parallel I/O interface designed for transfer of 8-bit, parallel, data between the Z-80/CPU and a variety of peripherals. The interfaces are TTL comparible and both ports are bidirectional.

3. The Z-80/CTC - This is a programmable Counter Timer Circuit with four independent channels, capable of providing counting and timing control functions for real-time events. The inputs and the outputs on this chip are also TTL compatible.

4. The Z-80/SIO - This is a programmable, dual-channel, serial I/O interface designed for communication with a variety of serial data peripherals. It is primarily a serial-to-parallel and parallel-to serial converter/controller using a number of different protocols.

5. The Z-80/DMA - This is a programmable Direct Memory Access Controller which can directly transfer data between the memory and the above-mentioned PIO and SIO interfaces in a cycle-stealing mode of operation.

Note - All of the above interfaces and controllers monitor the status of the peripherals. CPU polling is not involved. They also include priority interrupt systems which can be implemented by using the daisy chain logic technique. Additional external

logic is not required for this.

1-2.2 The Development System

ZILOG, INC. also provides a hardware/software development system for the Z-80 product line. Briefly, they include the following features:

1. The program development is RAM-based, not ROM-based.

2. The system is configured around a stand-alone, floppy disk.

3. The system provides real-time debugging capability.

4. The offline support software includes:

   Assembler
   Compiler
   Simulator, and
   Test pattern generation capability.

5. The resident software includes:

   RAM-based Assembler,
   RAM-based Text Editor,
   The BASIC Compiler,
   Real-time debugging program,
   Disk Operating System (DOS) with file maintenance, and
   ROM-based Executive firmware.

6. Diagnostic capability is also available. It includes:

   Rapid fault detection by means of system diagnostic software, and
   Continuous memory diagnostics performed in a background operation mode.

7. It is possible to replace RAM chips by ROM or PROM chips.

1-3 PRINCIPAL FEATURES OF THE Z-80 CPU

The main features of the CPU chip are presented below in summarized forms:

1. The Z-80 is a third generation μP which uses the N-channel, depletion-mode, silicon-gate technology.

2. A single, +5 volt ±5% power supply is used in the system.

3. A single-phase, TTL level, clock is used.

4. The Z-80 uses a 2.5 MHz clock. This gives an instruction execution time of 1.5 μs.

5. For a higher throughput rate, the Z-80 uses a 4.0 MHz clock which gives an instruction execution time of 1.0 μs.

6. The Z-80/CPU chip is available in a standard 40-pin, dual-in-line (DIP), package.

7. The Z-80 has an instruction set of 157 instructions which is software compatible with the 78 instructions of the INTEL 8080A μP.

8. The Z-80 instruction set includes instructions for:

   Bit manipulation,
   Byte manipulation,
   Character string operations, and
   Block transfers of data.

9. Two 16-bit Index Registers are available. They enhance the processing of look-up tables and arrays.

10. The Z-80 includes a duplicate set of general-purpose (GP) registers and status registers. The CPU has a total of 16 such registers.

11. The Z-80/CPU generates all the control signals for standard memory chips. Thus,

    Using only an external address decoder, static memory chips can be readily interfaced with the CPU,
    All controls for refreshing dynamic MOS memory chips are provided, and
    The control bus signals are capatible with the commonly-used, 18-pin and 22-pin 4K RAMs.

12. Expanded 16-bit arithmetic operations, as well as BCD operations, are available in the Z-80 CPU.

13. The Z-80 chips are designed to operate in temperature environments ranging from 0°C to +55°C.

1-4 THE CPU ARCHITECTURE

In the Z-80/CPU, there is a total of 22 registers/accumulators which are divided into two groups, namely, the general-purpose (GP) group and the special-purpose (SP) group which contains registers dedicated to performing only certain specific functions. Four of these registers are 16-bit units and the rest are 8-bit registers. The CPU has two accumulators. Additionally, the CPU has a 16-bit instruction register, with its associated instruction decoder, and the timing and control. All the above-mentioned registers are programmable, i.e., they can be accessed by the programmer.

1-4.1.1 GP Registers, Accumulators and Status Flag Registers

1. There is a total of 16, 8-bit, GP registers. Two of these are accumulators and two are status flag registers, assocated with each of the two accumulators.

2. These 16 registers are divided into two sets, the main set and the alternate set as shown in Figure 1-1.

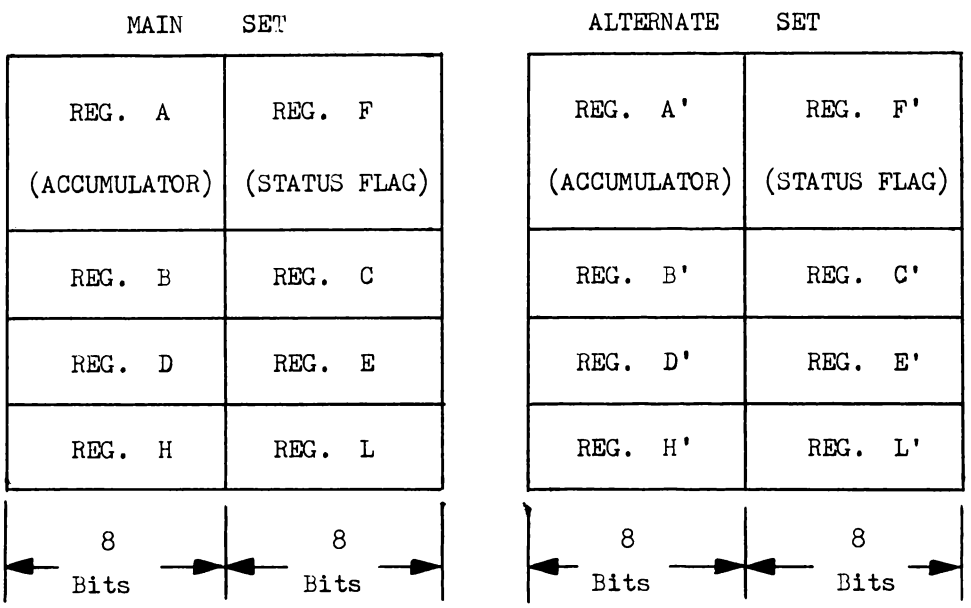


Fig. 1-1 General-purpose registers in the Z-80/CPU.

3. The Main Register Set units are labelled A (for the accumulator) and F (for its corresponding flag register). The rest are labelled B, C, D, E, H and L. Since registers A and F perform specific functions, it is more appropriate to say that the true GP registers are B, C, D, E, H and L registers only.

4. The second set is called the Alternate Register Set and is labelled in exactly the same manner but with a prime attached to each letter symbol. Once again B', C', D', H' and L' are the true GP register while A' and F' are the accumulator and the status flag registers respectively.

5. Accumulator A is the primary accumulator. Whenever a PUSH or a POP operation is performed involving the accumulator and the flag register, both registers are always transferred.

6. The remaining six registers, in both the main and the alternate set, can be used either individually as 8-bit units or they can be paired and used as 16-bit units as shown below:

Main Set	Alternate Set
B with C	B' with C'
D with E	D' with E'
H with L	H' with L'

7. The register pairs BC, DE,HL and B'C', D'E', H'L' can be used as secondary registers and/or data counters.

8. Under program control, either the main set or the alternate set can be selected and operated by means of a single instruction. Only one set can be used at any one time, not both.

1-4.1.2 Special-Purpose Registers

1. The Z-80/CPU has 6 special-purpose registers (SPR).

2. There are two 8-bit registers and four 16-bit registers as shown in Figure 1-2.

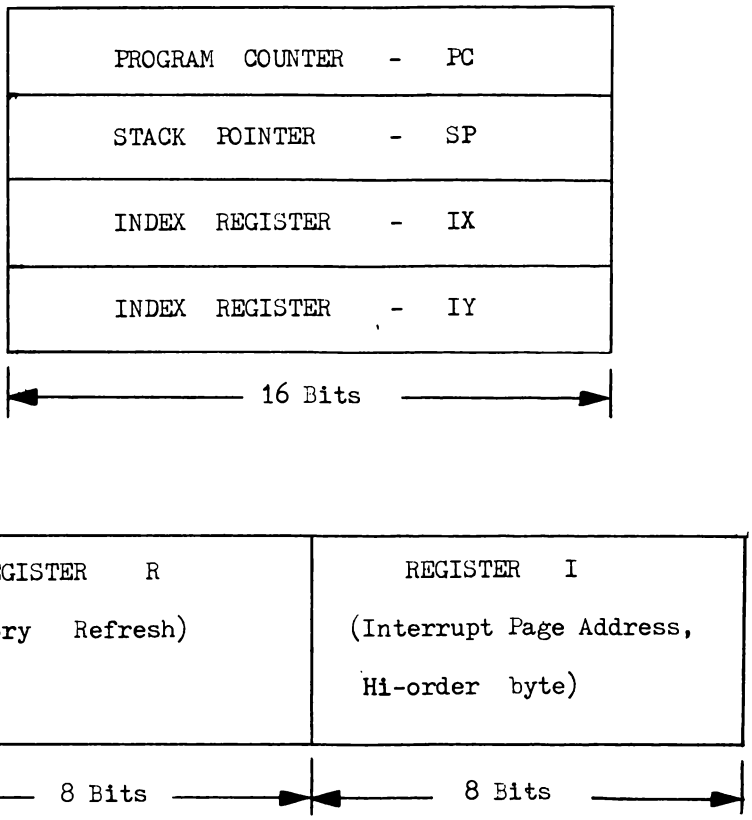


Fig. 1-2 Special-purpose registers in the Z-80/CPU.

3. A brief explanation of each SPR register follows:

   A. Program Counter (PC) - At any time, the PC holds the 16-bit address of the current instruction fetched from the program memory.

      After the contents of the PC (i.e., the address of the instruction) are placed on the address bus, the PC is automatically incremented.

      When a JUMP or a BRANCH is executed, the new address is automatically inserted in the PC.

   B. Stack Pointer (SP) - The Z-80 uses a pointer-type Stack to provide return addresses from BRANCH operations.

a previously-designated portion of the external RAM memory is used for the stack.

The stack is organized on a last-in, first-out (LIFO) basis.

By means of PUSH/POP instructions, data can be pushed on to the stack from any specified CPU register. Likewise, data can be popped from the stack into any specified CPU register.

The stack provides unlimited nesting capability.

Capability for multiple-level interrupts is also provided.

C. Index Registers (IX and IY) - These are two completely independent index registers.

Their primary function is to provide a 16-bit base address in the Indexed Addressing Mode.

They are extensively used in table look-up operations when locations in the memory are to be accessed.

For indexed addressing, a displacement byte is included in the instruction. The displacement is a signed twos complement integer and it is added to the base address in the index register to obtain the true or effective address.

D. Memory Refresh Register (R) - The Z-80 is one of the μPs which provides for the use of both static and dynamic MOS RAM chips in its system.

Dynamic memory chips are relatively inexpensive but can hold the information in them for only a very short time.

Thus, at millisecond intervals, the contents of each memory location are read out and rewritten in the same memory location.

The above process is called dynamic memory refreshing.

A counter is needed to track and update each memory address as each location is refreshed.

In the Z-80/CPU, the R register, which is a counter, performs the address incrementing function.

The memory refreshing operation is completely transparent to the programmer or the user.

Although the R register is not used for operations other than refreshing, it is programmable, i.e., the programmer can externally load the register for testing purposes.

E. Interrupt Page Address Register (I) - In response to an interrupt, the Z-80 provides the capability to branch to a service subroutine by means of an Indirect Addressing mode as an optional addressing mode.

Indirect addressing requires that the memory page address be provided as well as the local address in that particular page.

The I register is used for storing the high-order byte (i.e., the page address) of the 16-bit starting address of the subroutine.

The local address (i.e., the low-order byte) is provided by the interrupting device itself.

The I register is also called the Interrupt Vector Register.

1-4.2 The Arithmetic Logic Unit (ALU) -

1. The ALU normally performs arithmetic and logic operations on 8-bit operands. Additionally, the ALU can also perform additions and subtractions on 16-bit operands located in the 16-bit registers.

2. The ALU communicates with the other internal registers of the CPU by means of the 8-bit internal data bus.

3. The ALU performs the following arithmetic functions:

   ADD
   SUBTRACT
   INCREMENT
   DECREMENT

4. The ALU performs the following logical functions:

   AND
   OR
   EXCLUSIVE-OR
   COMPARE
   RIGHT ROTATE
   LEFT ROTATE
   RIGHT SHIFT LOGICAL
   LEFT SHIFT LOGICAL
   RIGHT SHIFT ARITHMETIC
   LEFT SHIFT ARITHMETIC

5. The ALU performs the following additional functions:

   Test or sample bit
   Set bit (to a binary one)
   Reset bit (to a binary zero)

1-4.3 Instruction Register, Decode and Control

1. Figure 1-3 shows a generalized block diagram of the Z-80/CPU.

2. When the instruction is fetched from the program memory, it is loaded into the 16-bit instruction register.

3. The instruction is then decoded and various control signals are sent to the CPU and external devices of the system as commanded by the particular instruction.

1-4.4 Program Status Word (F and F')

From Fig. 1-1 we see that each accumulator, A and A', has an associated status flag register, F and F'. The function of each of these two registers is to flag out and indicate the current status of certain conditions in its associated accumulator resulting from either an arithmetic or logical operation. Although both registers, F and F', are 8 bits long, only 6 of these bits are used in the Z-80. The format of these two registers is shown in Fig. 1-4. The two unused bit positions are shown as X. Also remember that these two registers are made up of flip-flops which can be set to the 1 side or reset to the 0 side. A brief explanation of each status flag is given below:

1. Sign Bit (S) - The sign bit flag is used in conjunction with arithmetic operations that use signed numbers (a 1 for negative numbers and a 0 for positive numbers). The most-significant-bit (MSB) of the number is the sign bit. This means that the S flag bit indicates the status of the least significant 7 bits in the accumulator associated with the status flag register. The S flag is set if the MSB of the accumulator is a 1 and reset if it is a 0.

2. Zero Bit (Z) - The zero flag checks all the bits in the accumulator which results from either an arithmetic or a logical operation. If all the bits in the associated accumulator are zero, the Z flag is set to the 1 side; otherwise, it is reset to the 0 side.

3. Half Carry Bit (H) - Binary-coded decimal (BCD) arithmetic can be used in the

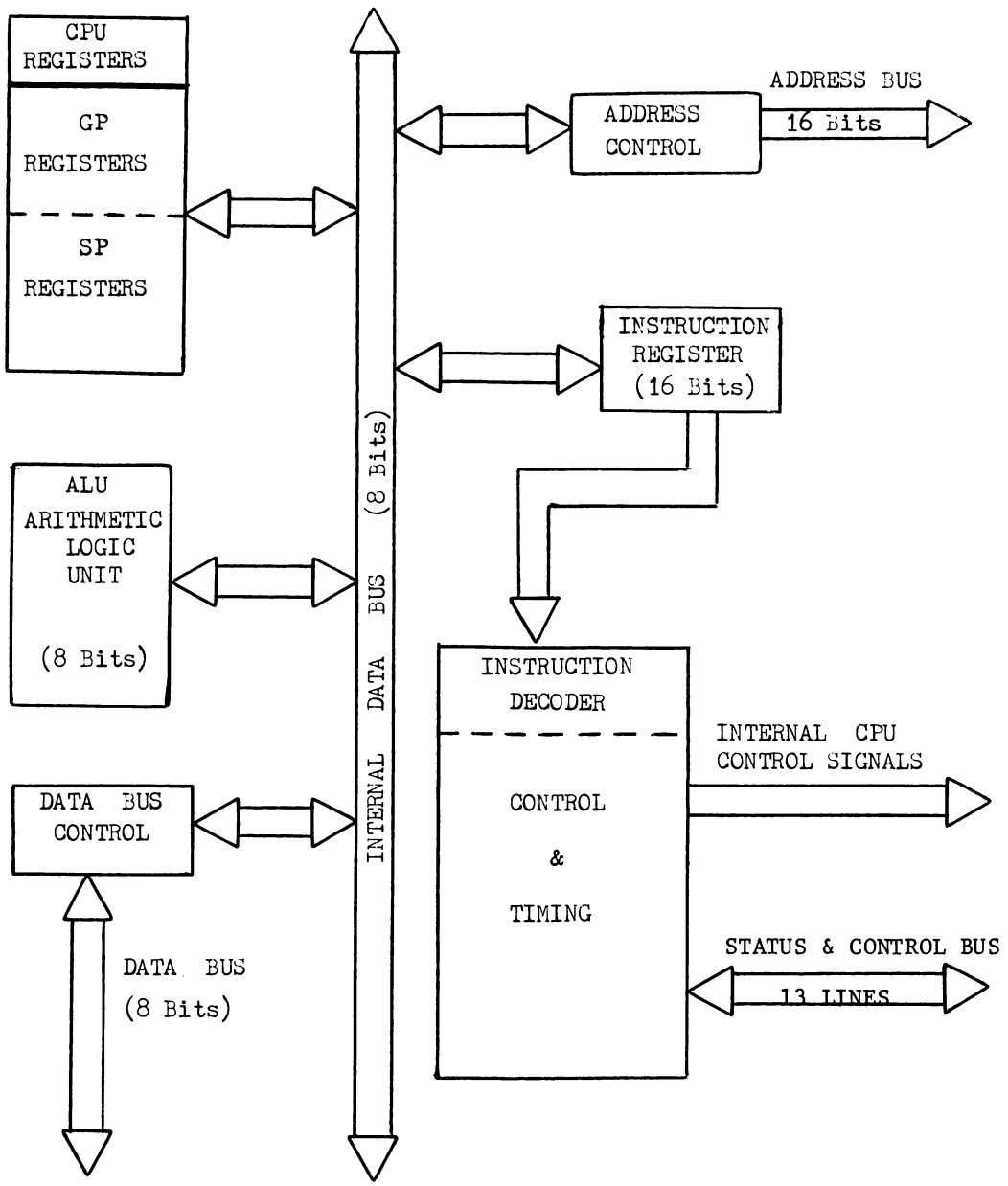


Fig. 1-3 Generalized block diagram of the Z-80/CPU.

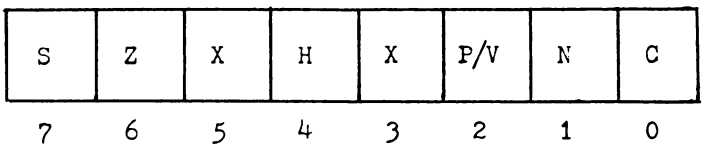


Fig. 1-4 Program status word format.

Z-80. Because of the nature of BCD additions and subtractions, a BCD carry or borrow could result from arithmetic operations on the least significant 4 bits of the operands. Consequently a decimal arithmetic adjust (DAA) operation must be performed to produce a correct BCD answer. The H flag is set when such a DAA operation is required. Note that the H flag is also referred to as the auxiliary carry flag and is indicated by the symbol A_C. The H flag is never tested by conditional JUMP or BRANCH instructions.

4. Parity/Overflow Bit (P/V) - This bit is used for two separate functions in the Z-80/CPU. When used as a parity bit, it is set if the result of a logical operation (such as AND, OR, XOR) results in an even number of ones in the accumulator. Also, during one specific input operation, i.e. IN r, (C), the parity of the incoming data byte is indicated by this flag. Other normal input operations do not affect this flag. As an overflow bit, this flag is used when signed twos complement arithmetic is performed. Since the MSB of the accumulator is the sign bit, this flag is set when the resulting quantity in the accumulator exceeds the maximum possible number +127, or is less than the minimum number -128.

5. Subtract Bit (N) - The DAA operations, required in BCD arithmetic, are performed by two different algorithms; one for the addition and the other for the subtraction process. The N flag indicates which instruction was executed that required the DAA operation, addition or subtraction, so that the appropriate algorithm may be selected and executed. The flag is set to the 1 side if a subtraction is involved.

6. Carry Bit (C) - If a carry is generated from the MSB of the accumulator, the C flag is set. This flag is set if a carry is generated as a result of an addition process and also if a borrow is generated because of a subtraction process. Also, in shift and rotate operations involving the accumulator, this flip-flop can be included as a link flip-flop under program control.

7. X Bits - Bits in positions 3 and 5 of the status word, which are indicated by X in Fig. 1-4, are ignored. They are not used in any operations.

1-5 ADDRESSING MODES

The Z-80 system offers a wide variety of addressing modes, most of which have been covered in considerable detail in Chapter 5 of the main hardcover textbook. Consequently, only a brief explanation of each, as used in the Z-80/CPU, is given below.

1-5.1 Implied Addressing

This is a very simple mode of addressing where the OP code within the instruction itself implies one or more of the CPU registers as either the source or the destination of the operand(s). A typical example of this addressing mode is that of arithmetic operations in which the destination of the result is always automatically implied as one of the two accumulators.

1-5.2 Register Addressing

The Z-80/CPU architecture provides the capability to address specific registers in the CPU which can be used for certain operations. The register(s) concerned is/are identified by certain bits of information in the OP codes themselves. An example of this mode of addressing is an instruction which would load (i.e. transfer) the contents of register D into register E.

1-5.3 Register Indirect Addressing

In this addressing mode, a 16-bit pointer, which is the ultimate address of the memory location to be accessed, is stored in one of the register pairs in the CPU; usually the HL pair. The specific register pair of the CPU is addressed by certain bits in the OP code of the instruction which could be either one or two bytes. This addressing mode is very powerful. Very often this mode is used to access 16-bit operands in the memory which are generally stored as two bytes in two successive memory locations. In such a situation the CPU register pair (such as the HL pair) points to the location of the low-order byte of the desired operand. The high-order byte is then accessed simply by incrementing the contents of the register pair which would be the HL pair in this case.

1-5.4 Bit Addressing

Several instructions in the Z-80 enable the programmer to set, reset and test or sample any bits of words located in any of the CPU registers or any locations in the memory. These bit operations are performed by specifying the location of the word by means of one of the three addressing modes, namely, Register Addressing (Sec. 1-5.2), Register Indirect Addressing (Sec. 1-5.3) and Indexed Addressing which will be covered in Sec. 1-5.8. The particular bit to be manipulated upon in the specified word is identified by a 3-bit field in the OP code of instruction.

1-5.5 Immediate Addressing

In this addressing mode, the operand is not located in any CPU register or memory location. Instead, it is contained directly within the instruction itself. The instruction could be either a 2-byte or a 3-byte instruction with either a single-byte or a 2-byte OP code. The 8-bit operand is contained in the following byte.

1-5.6 Immediate Extended Addressing

This mode is similar to the immediate addressing mode described in Sec. 1-5.5. The only difference is that the OP code (either one or two bytes) is followed by a 2-byte operand. The first byte of the operand is the low-order byte and the second byte is the high-order byte. Thus, the instruction could be either 3 or 4 bytes long.

1-5.7 Extended Addressing (Direct Addressing Mode)

This addressing mode involves the standard direct addressing mode. The OP code can be either one or two bytes long. The address field always contains two bytes. In this addressing mode, the two address bytes of the instruction contain the true or effective memory address where the operand is located or is to be stored. By means of the extended addressing mode, all locations in the memory can be addressed directly.

1-5.8 Indexed Addressing

This is the standard indexed addressing mode whose operation is explained in detail in Sec. 5-7 of the main hardcover textbook. The OP code of the instruction consists of two bytes. The address field is a single byte and it contains the displacement which, when added to the contents of the previously-loaded index register, gives the true or effective address of the memory location to be accessed. In some cases the instruction could have a fourth byte if an immediate operand is involved in the operation specified by the instruction.

1-5.9 Relative Addressing

A 2-byte instruction is used for the Program Counter (PC) Relative Addressing Mode. The first byte contains the OP code and the second byte an 8-bit, signed, 2s complement displacement. This displacement is added to the address of the next instruction's OP code which is located in the program counter. Since the instruction using the relative addressing mode itself is two bytes long, the displacement is added to the contents of the PC incremented twice. Recall from previous discussion of the Relative Addressing mode in the main hardcover textbook (Sec. 5-5) that this method normally

allows access to an address range from +127 to -128 relative to the address of the instruction contained in the program counter. However, in the Z-80/CPU, this is not the case. Because the signed displacement is added to PC + 2, the accessible memory address ranges from +129 to -126.

1-5.10 Modifies Page Zero Addressing

A modified version of the Base Page or Page 0 addressing mode, which was described in Sec. 5-4.2 and Sec. 5-4.3 of the main hardcover textbook, is available in the Z-80. This mode is initiated by a single-byte CALL instruction (often referred to as a restart instruction) which enables access to 8 locations in Page 0 of the memory. These 8 locations contain 16-bit pointers which are addresses in the memory. This addressing mode is used to BRANCH to frequently-used subroutines having 16-bit starting addresses by means of a single-byte instruction. It operates as explained below.

1. The instruction contains a 3-bit code which specifies the address of one of the 8 reserved locations in Page 0 of the memory.

2. The 8 locations in Page 0 are previously loaded with the 16-bit starting addresses of the frequently-used subroutines.

3. The incremented contents of the program counter are pushed into the stack.

4. The 16-bit address in the accessed location of Page 0 is then loaded into the program counter (at the programmer's option).

The above-mentioned application of these reserved locations is frequently used but is not the only option available to the programmer. These locations can also be used for other purposes. For instance they may contain the actual subroutine if it is short one. Note that this mode of addressing is really a combination of the Page 0 Relative and the Extended Indirect Addressing modes.

1-6 PACKAGE PIN ASSIGNMENTS AND FUNCTIONS

The Z-80/CPU chip is available in a standard 40-pin, dual-inline-package (DIP). The package pins and their respective functions, together with their symbolic designations are presented in congenient tabular form in Tables 1-1 through 1-6. Functionally, the pins can be divided into five groups as shown in Fig. 1-5. Note that some lines are shown as unidirectional and others as bidirectional. In the latter group some of the lines may indicate signal flow in one direction and other in the other direction. The tables clarify this with the following arrow symbols:

CPU ⟶ Indicates signals flowing from the CPU.
CPU ⟵ Indicates signals flowing into the CPU.
CPU ⟷ Indicates bidirectional signal flow on the same line.

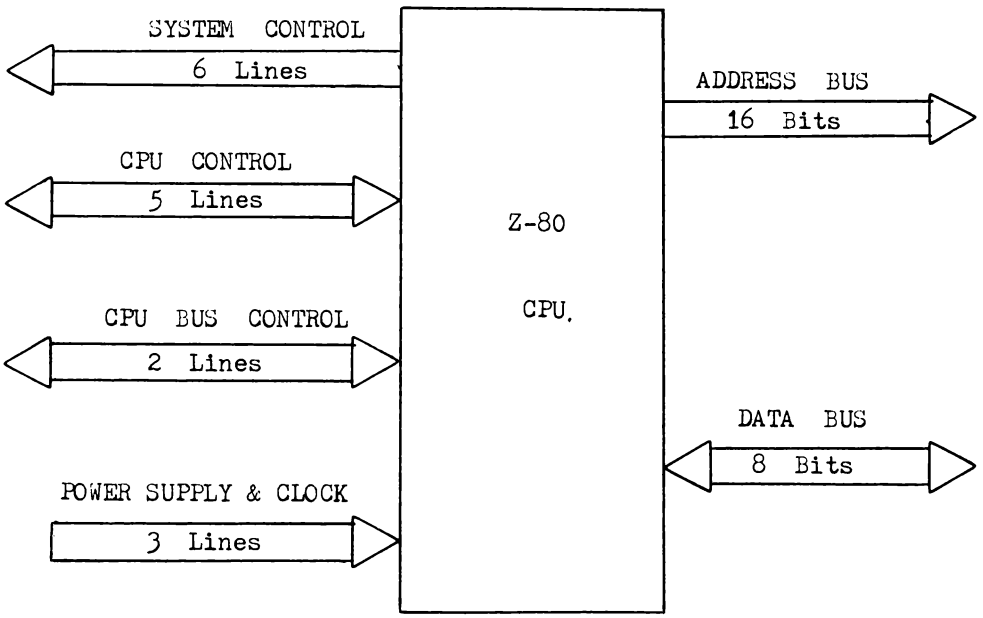


Fig. 1-5 Diagram of CPU package pin groups.

Table 1-1 Address and data busses groups.

PIN DESIGNATIONS	SIGNAL FLOW DIRECTIONS	FUNCTIONS
A0 - A15	CPU ⟶	16-bit, tristate, address bus, is active high. Provides direct addressing capability for 64K byte memory and I/O devices for data transfers. Upto 256 input and 256 output ports may be selected by the lower 8 bits. A0 is LSB and A15 is MSB.
D0 - D7	CPU ⟷	8-bit, tristate, data I/O bus is active high. It is used for data transfers between the CPU and the memory as well as between the CPU and the I/O devices.

Table 1-2 System control group

PIN DESIGNATIONS	SIGNAL FLOW DIRECTIONS	FUNCTIONS
$\overline{M1}$	CPU ⟶	Output signal indicating that the current machine cycle involves OP code fetch subcycle. If the OP code contains 2 bytes, this signal is generated as each byte is fetched. When this signal is active with $\overline{IORQ}$ signal, it indicates an interrupt acknowledge cycle. It is active low.
$\overline{MREQ}$	CPU ⟶	Tristate, memory output request signal indicating that the bits on the address bus are valid memory address. It is active low.
$\overline{IORQ}$	CPU ⟶	Tristate, I/O request signal indicates that the low byte of the address bus has a valid I/O address for I/O data transfers. When interrupt is acknowledged, this signal is simultaneously generated with $\overline{M1}$ indicating that an interrupt vector can be placed on the data bus. It is active low.
$\overline{RD}$	CPU ⟶	Tristate signal indicating that the CPU is ready to read data outputted by either the memory or the I/O device. Upon receiving this signal, the memory or the I/O device gates the data on to the data bus. It is active low.
$\overline{WR}$	CPU ⟶	Tristate signal to the memory or I/O device indicating that the CPU has placed valid data on the data bus for acceptance by the memory or the I/O device.
$\overline{RFSH}$	CPU ⟶	Indicates that lower 7 bits of the address bus contain a refresh address for dynamic memories. Thus, the current $\overline{MREQ}$ signal executes a refresh read for all such memory chips.

Table 1-3 CPU control group.

PIN DESIGNATIONS	SIGNAL FLOW DIRECTIONS	FUNCTIONS
$\overline{\text{HALT}}$	CPU ⟶	Indicates that the CPU has executed a HALT instruction. CPU then resumes execution only upon receipt of an interrupt (either maskable or non-maskable). During HALT, CPU executes NOOP instructions so that information in dynamic memories can be maintained by refreshing.
$\overline{\text{WAIT}}$	CPU ⟵	An active low signal from either the memory or the I/O device indicating that it is not ready for a data read or write operation. The CPU idles until the memory or I/O indicates that it is ready for a transfer.
$\overline{\text{INT}}$	CPU ⟵	Interrupt Request signal, generated by the I/O devices. CPU completes current instruction and honors this signal only if the software-controlled interrupt F/F is enabled (IFF) and the $\overline{\text{BUSRQ}}$ signal is inactive. CPU sends out an acknowledge signal $\overline{\text{IORQ}}$ at the beginning of the next instruction cycle.
$\overline{\text{NMI}}$	CPU ⟵	This is a negative-going, edge-triggered, non-maskable interrupt request which has a higher priority than $\overline{\text{INT}}$ and can override the (IFF) which is software-controlled. The contents of the program counter are saved in the stack for return to the original point in the interrupted program.
$\overline{\text{RESET}}$	CPU ⟵	This signal forces the program counter to reset to zero. It initializes the CPU and disables the interrupt enable F/F. Also, all control output signals are deactivated and the address and data busses go into the high-impedance states.

Table 1-4 CPU bus control group.

PIN DESIGNATIONS	SIGNAL FLOW DIRECTIONS	FUNCTIONS
$\overline{\text{BUSRQ}}$	CPU ◄———	This signal puts the tristate address and data busses, as well as the tristate output control signal lines into the high-impedance state. The external devices then control these busses and lines. The CPU, of course, completes its current machine cycle and then complies with this request.
$\overline{\text{BUSAK}}$	CPU ———►	The CPU sends this acknowledge signal to the requesting external device that the address and data busses as well as the tristate output control lines are in the high-impedance state and are under control of the external device.

Table 1-5 Power supply and clock group.

PIN DESIGNATIONS	SIGNAL FLOW DIRECTIONS	FUNCTIONS
ø	CPU ◄———	This is the single-phase, TTL level clock input. A 330 ohm pull-up resistor to +5 volts is required.
+5V	CPU ◄———	This is the single power supply of +5 ±5 % that is needed for the Z-80.
GND	CPU ◄———	This is the power supply ground.

Table 1-6 shows the actual pin numbers assigned to each of the functional designation covered in Tables 1-1 through 1-6.

Table 1-6 Pin assignment numbers

PIN DESIGNATIONS	PIN NUMBERS	PIN DESIGNATIONS	PIN NUMBERS
Address Bus		System Control	
A0	30	$\overline{M1}$	27
A1	31	$\overline{MREQ}$	19
A2	32	$\overline{IORQ}$	20
A3	33	$\overline{RD}$	21
A4	34	$\overline{WR}$	22
A5	35	$\overline{RFSH}$	28
A6	36		
A7	37	CPU Control	
A8	38	$\overline{HALT}$	18
A9	39	$\overline{WAIT}$	24
A10	40	$\overline{INT}$	16
A11	1	$\overline{NMI}$	17
A12	2	$\overline{RESET}$	26
A13	3		
A14	4	CPU Bus Control	
A15	5	$\overline{BUSRQ}$	25
		$\overline{BUSAK}$	23
Data Bus			
D0	14	Power Supply and Clock	
D1	15		
D2	12	Ø	6
D3	8	+5V	11
D4	7	GND	29
D5	9		
D6	10		
D7	13		

1-7 REVIEW QUESTIONS

1-1 The Z-80 μP is simlar to some of the currently available μPs on the market in many respects. Circle the correct one(s).

a. The Z-8000 by ZILOG
b. The 8086 by INTEL
c. The 8080Z by INTEL
d. The 6800 by MOTOROLA

1-2 The following statements apply to the Z-80 μC system. Indicate true or false.

a. ____The Z-80/CPU is a third generation μP.

b. ____The Z-80 uses ± 5% power supply.

c. ____An instruction execution time of 1.0 μs is obtained with a 2.5 MHz clock.

d. ____The Z-80/CPU is available in a 28-pin DIP package.

1-3 The following statements relate to the instructions available for the Z-80. Indicate true or false.

a. ____Byte manipulation is possible.

b. ____Bit manipulation capability is not available.

c. ____Block of data can be transferred.

d. ____Character String operations are available.

1-4 Fill in the blanks.

a. Index Registers are primarily used for _____ and _____ Operations. (Array; direct addressing; table look-ups; JUMP; BRANCH).

b. There is/are _____index registers in the Z-80. (1; 2; 4; 8; 16)

c. The index register(s) is/are _____ bits long. (4; 8; 12; 16)

1-5 Name one technique that can be used for priority interrupts in the Z-80 (at the user's option).

1-6 The Z-80 uses a duplicate set of GP registers. This is done to provide: (Circle all that apply)

a. Capability to handle both 8 and 16 bit operands.

b. Faster memory data transfers.

c. Faster interrupt processing capability.

d. Faster I/O data transfers.

1-7 Indicate true or fase for the Z-80.

a. ___ Operates in a temperature range of -55°C to +55°C.

b. ___ BCD arithmetic operations are available.

c. ___ Both static and dynamic memory chips can be used.

d. ___ A total of 8 GP registers are available.

1-8 The Alternate Register set in the Z-80 includes _____ registers, each _____ bits long. (4; 6; 8; 12; 16; 32).

1-9 ___ How many bits are used in the Status Flag Register of the Main Register Set? (2; 4; 6; 8; 16)

1-10 The following questions relate to the GP registers in the Z-80. Answer them yes or no.

a. ___ Is it possible to use register A' and F' as a 16-bit unit?

b. ___ Is it possible to pair register D and D' as a 16-bit unit?

c. ___ Is it possible to use Register C as a status register for Register B?

1-11 Assume that the main Register set is used during a certain program and an interrupt signal comes in. Indicate true or false for the following statements for the time during which a fast interrupt subroutine is serviced.

a. ___ The contents of all the alternate set registers are automatically transferred into the registers of the main set.

b. ___ The alternate set's registers are used by the interrupt service subroutine.

c. ___ The interrupt service routine uses the registers from both the main and the alternate sets.

d. ___ Status saving may be done in the alternate set.

1-12 The Z-80/CPU has _____ SP registers. (4; 6; 8; 10; 12; 16)

1-13 In the Z-80/CPU, the program counter is incremented (circle the correct statement)

a. Before the contents of the PC are on the address bus

b. After the contents of the PC are on the address bus.

1-14 Indicate true or false for the Z-80/CPU SP registers.

a. ___ A LIFO cascade stack is used.

b. ___ The stack provides unlimited nesting capability.

c. ___ Multiple level interrupts are possible.

d. ___ Registers IX and IY are completely independent of each other.

1-15 Which of the following SP registers is used for keeping track of the addresses during the refreshing of dynamic RAM chips? (Circle the correct one)

a. Stack Pointer
b. Program Counter
c. Register I
d. Register R
e. Register IX
f. Register IY

1-16 The following statements apply to interrupt processing and Register I. Indicate true or false.

a. ___ Register I is used for directly branching to the interrupt service subroutine.

b. ___ Register I stores the entire 16-bit starting address of the interrupt service subroutine.

c. ___ The local address of the interrupt service subroutine is provided by the interrupting device itself.

d. ___ The page address of the interrupt service subroutine's starting address is stored in the I register.

1-17 Circle the statement(s) below which apply to the Z (zero) flag bit in the F and F' registers.

a. The Z flag checks all the GP registers for zeros.
b. The Z flag checks all the bits of its corresponding accumulator.
c. If all the bits of the accumulator are 0 the Z F/F is set to the 1 side.

1-18 When is the sign bit flag (S) in the status register set? Circle the correct answer(s).

a. When the MSB of the accumulator indicates a positive number.
b. When any of the GP register of that particular set is a negative quantity.
c. When the MSB of the corresponding accumulator is a 1.
d. When the LSB of the corresponding accumulator is a 1.

1-19 The statements below apply to the Carry Bit (C) in the F and F' registers. Indicate true or false.

a. ___ C is set if there is no carry from an arithmetic operation.

b. ___ C is set if a borrow results from a subtraction operation.

c. ___ C is set if a carry is generated from the MSB position of the accumulator.

1-20 Fill in the blanks.

The Half Carry Bit (H) is _____ to indicate the necessary correction for the carry/borrow resulting in the least significant 4 bits from a _____ arithmetic operation. (BCD; set; reset; HEX; binary; octal)

1-21 The following statements apply to the DAA operations and the Subtract Bit flag (N). Indicate true or false.

a. ___ In DAA operations, the same algorithm is used for addition and subtraction processes.

b. ___ The N flag is set if the prior instruction executed were a subtraction instruction.

1-22 The following statements apply to the P/V flag bits. Indicate true or false.

a. ___ The P flag is set if the result in the accumulator contains an even number of 1 bits.

b. ___ The overflow flag, V, is used when a logical function is performed and the result stored in the accumulator.

c. ___ The V flag is set when the resulting quantity in the accumulator exceeds +127.

d. ___ The V flag is set when the resulting quantity in the accumulator is less than -128.

1-23 The questions below apply to the Z-80 addressing modes. Answer them.

a. _____ Is it always necessary to add second byte after the OP code if Register B is to be accessed?

b. _____ Refer to (a) above. Which addressing mode will be used in this situation?

c. _____ Which addressing mode contains bits in the OP code to identify the accumulator as the destination?

1-24 Which addressing mode in the Z-80/CPU will allow a 2-byte operand in the memory to be accessed by means of a single-byte address in the instruction.

_____ .

1-25 Refer to 1-24 above. Where is the pointer for the 2-byte memory operand stored?

_____ .

1-26 Refer to 1-24 and 1-25 above. Do we need a second instruction to access the second byte in the memory? If not, why not?

_____

_____

_____ .

1-27 We want to reset a certain bit in a memory location. Indicate true or false for the situation.

a. ___ The bit is identified by a separate byte in the OP code.

b. ___ The bit is identified by a 3-bit field in the OP code.

c. ___ Register Indirect Addressing cannot be used.

d. ___ Indexed addressing can be used.

e. ___ The bit in the memory location can also be set.

1-28 The following statements relate to Immediate and Immediate Extended addressing modes. Indicate true or false.

a. ___ The operand is not located in a CPU register.

b. ___ The operand is located in the memory.

c. ___ The operand is located within the instruction itself.

d. ___ The instruction is either 3 or 4 bytes long when the Immediate Extended Addressing Mode is used.

1-29 The following statements relate to the Extended Direct/Indirect Addressing Modes. Fill in the blanks.

a. The OP code can be either _____ or _____ bytes long. (1; 2; 3)

b. The address field always contains _____ byte(s). (1; 2; 3)

c. In Direct Addressing, the _____ byte(s) contains(s) the true or effective address of the memory location. (1; 2; 3)

d. In Direct Addressing, the _____ order byte follows the OP code. (High; low)

e. In Indirect Addressing, the address field contains the _____ bit pointer. (8; 16)

1-30 The following statements relate to the Relative Addressing Mode in the Z-80/CPU. Indicate true or false.

a. ___ A single-byte instruction is used for this mode.

b. ___ The second byte of the instruction contains a signed twos complement displacement.

c. ___ The displacement is added to the address of the next instruction's OP code in the program counter.

d. ___ The displacement is added to the contents of the PC which is incremented once.

e. ___ The accessible memory addresses range from +129 to -126 relative to the address of the instruction.

1-31 When the Indexed Addressed Mode is used, the OP code of the instruction contains ___ byte(s) and the address field contains ___ byte(s). (1; 2; 3)

1-32 Indicate true or false for the Modified Page Zero Addressing Mode.

a. ___ Pointers from memory addresses are stored in the first 8 locations of Page 0

b. ___ The pointers are 16 bits long.

c. ___ 16-bit starting addresses of subroutines can be reached by means of a single byte.

d. ___ The Page 0 locations are identified by a 4-bit code in the OP Code field.

e. ___ It is a combination of the Page 0 Relative and the Extended Indirect Addressing Modes.

2 THE Z-80 INSTRUCTION SET (PART ONE)

CHAPTER OBJECTIVES

The following are the objectives for this chapter:

1. To briefly describe the 8 fundamental categories of instructions in the instruction set.

2. To present the generalized instruction formats of the single, 2, 3 and 4 byte instructions,

3. To describe the mnemonic structure of the instructions, including the fixed and the variable parts, and the situation with implied operands.

4. To present and explain the symbolic expressions of operations and functions,

5. To present and explain the status register symbols,

6. To present, in convenient tabular form, the mnemonics, symbolic operations, status flags, OP CODES, the number of bytes and operation descriptions of the following groups of instructions:

   (a) 8-bit and 16-bit LOAD instructions,
   (b) 16-bit STACK PUSH/POP instructions,
   (c) EXCHANGE instructions,
   (d) BLOCK TRANSFER AND SEARCH instructions,
   (e) 8-bit arithmetic and logic instructions, and
   (f) 16-bit arithmetic instructions.

TEXTBOOK REFERENCES

For reviewing material in the main hardcover textbook, relevant to the topics covered in this chapter, the following chapter and/or pertinent sections are suggested:

1. For a general review of instructions - Chapter 4

2. For categorization of instructions - Sec. 4-2.1

3. For data transfer instructions - Sec. 4-2.2.1
4. For stack operations and instructions - Sec. 3-5.2.2
5. For arithmetic instructions - Sec. 4-2.2.2
6. For logic instructions - Sec. 4-2.2.3
7. For symbolic expressions and mnemonics - Sec. 4-3 and Sec. 4-4

2-1 INTRODUCTION

The INTEL 8080A μP has a set of 78 instructions. Because the Z-80 is an enhancement of the 8080A, the Z-80 instruction set includes all the instructions of the 8080A. An additional 79 instructions make up a total of 157 instructions. For an 8-bit μP, the Z-80 has a very powerful instruction set. The instructions are divided into 8 convenient categories which are briefly described below.

2-2 CATEGORIES OF INSTRUCTIONS

2-2.1 The Load and Exchange Group

This group of instructions primarily involves transfers of 8-bit and 16-bit words between either the internal registers of the CPU, or, between the CPU registers and the external memory. The instruction specifies both the source of the word as well as the destination involved in each transfer. Note that such transfers do not destroy the contents of the source location. The exchange instructions swap the contents of the two specified locations. This group also includes load immediate instructions which load the operand, contained in the instructions themselves, into one of the specified CPU registers or the specified location in the external memory.

2-2.2 The Block Transfer and Search Group

This group of instructions makes it possible to transfer a block of data (containing any number of words) from one location in the memory to another location in the memory by means of just one instruction. The block search group of instructions enables the user to search an entire block of data (of any length) in the memory for an 8-bit word or character with just one instruction. When the desired word is found, or when the end of the block is reached, the instruction and the search are automatically terminated.

2-2.3 The Arithmetic and Logic Group

This group of instructions performs various logical and arithmetic operations on operands. The words to be operated upon can be located either in external memory, the accumulator or the GP registers of the CPU. In the Z-80, the result of any of these operations is always placed in the accumulator. Depending upon the result of the operation, the appropriate flag is then set in the appropriate Status Register, F or F'. Also included in this group are instructions capable of performing 16-bit additions and subtractions on operands located in 16-bit registers.

2-2.4 The Rotate and Shift Group

This group of instructions make it possigle to perform right and left shifts as well as right and left rotates on any operands located anywhere in the external memory or in the CPU registers. Right and left rotations can be made either with or without including the Carry flip-flop. Both logical and arithmetic shifts can be performed. In logical shifts all bits in the register are given the same treatment. In arithmetic shifts, the sign bit in the most-significant-bit position is preserved and not altered during the shifting operation.

2-2.5 The Bit Manipulation Group (Set, Reset and Test)

This group of instructions makes it possible to set, reset or test any bit in any external memory location, the accumulator or any CPU register with a single instruction. These instructions are particularly useful in control applications.

2-2.6 The JUMP, CALL and RETURN Group

This group contains instructions for performing JUMPS, CALLS (i.e. BRANCHES) and RETURNS (from BRANCHES). Basically, these instructions are used for transferring control of the CPU from one part of the program to another (JUMPS) or between the main program and subroutines (BRANCHES and RETURNS).

2-2.7 The Input/Output Group

This group of instructions allows the user to perform a variety of different data transfers between the I/O devices and the CPU registers or between the I/O devices and the external memory. With just one instruction, it is possible to transfer a single block of data (maximum 256 bytes) directly between an I/O device and the memory. Some instructions in this group will affect some of the flags in the status register (depending upon the conditions involved) so that additional operations are not required to determine the status of the transferred data.

2-2.8 The Basic CPU Instructions Group

This group contains instructions which are used to define and control various modes and options available in the CPU. These include setting the mode interrupt response, setting or resetting the interrupt flip-flop, decimal arithmetic adjust (DAA), setting the Carry flip-flop, the NOP instruction, and other similar operations.

2-3 INSTRUCTION FORMAT AND SYMBOLS

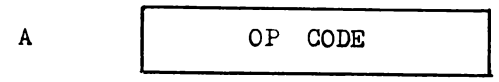
2-3.1 The Generalized Instruction Format

A Z-80 instruction could be made up of 1, 2, 3 or 4 bytes. The OP code portion of the instruction may consist of either 1 or 2 bytes. Depending upon the instruction involved, the OP code is followed by either a 1 or 2-byte address. If the instruction happens to be an Immediate type instruction, then the OP code is followed by a 1- or 2-byte operand. All the bytes of any multi-byte instruction are stored in consecutive memory locations. Figure 2-1 shows some possible combinations of multi-byte instructions where the starting address of the OP code is identified as A.

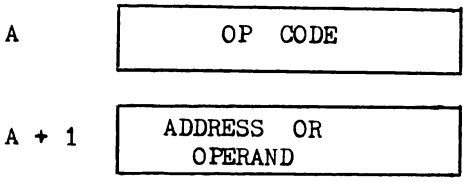
2-3.2 The OP Code

As shown in Fig. 2-1, the OP code of the Z-80 instruction can be either one or two bytes. Using the Implied Addressing Mode, data words can be transferred between the internal registers of the CPU. This means that the instruction itself must contain the codes, identifying addresses, for both the source and the destination registers.

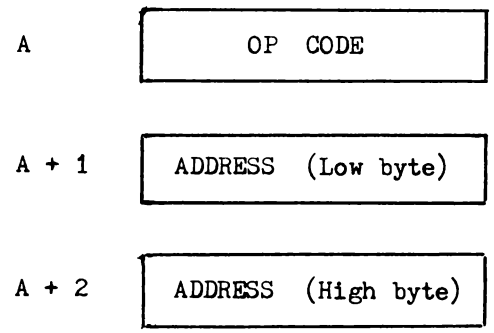
In the OP code byte, the destination register is designated as r_1 and the source register as r_2. The 3-bit identifying codes or addresses, assigned to the 7 CP registers are shown below in Table 2-1.



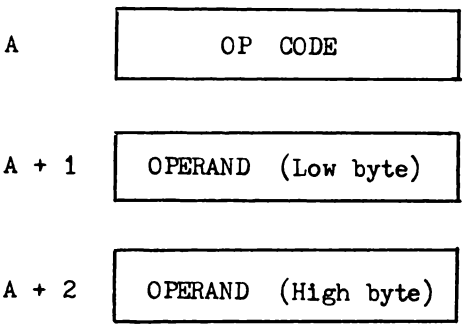
a. Single-byte instruction.



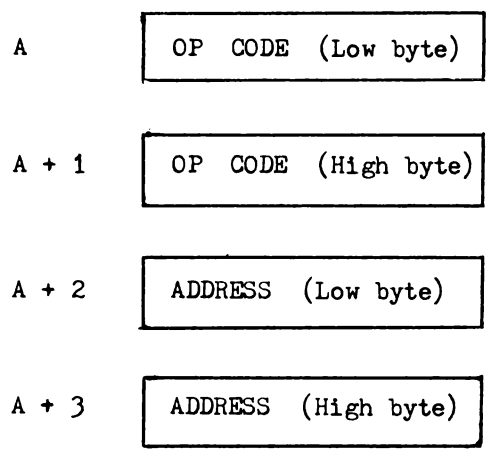
b. 2-byte instruction.



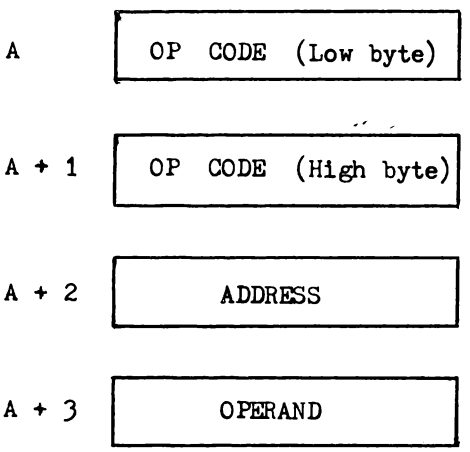
c. 3-byte instruction.



d. 3-byte instruction.



e. 4-byte instruction.



f. 4-byte instruction.

Fig. 2-1 Generalized Z-80 instruction formats.

Table 2-1 Address codes of the internal GP register

Register	Code
A	111
B	000
C	001
D	010
E	011
H	100
L	101

The relative placement of these codes in the OP code determines which register is the source and which is the destination. The instruction shown in Fig. 2-2 explains this.

Suppose we wish to transfer the contents of register D of the main set into register E of the main set. The single-byte instruction is then formatted as shown in Fig. 2-2.

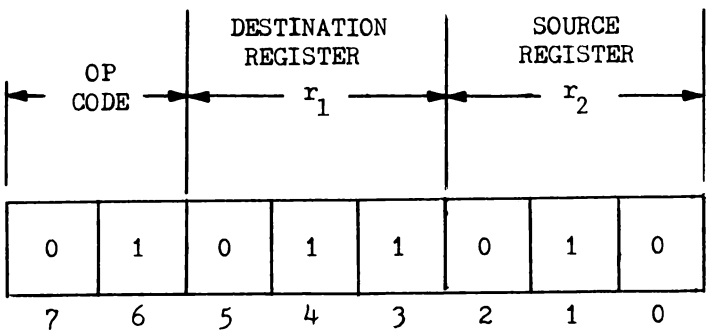


Fig. 2-2 Instruction format for internal CPU register transfer.

The 01 combination in bit positions 7 and 6 respectively is the register-to-register transfer OP code. Bit combination 010 in bit positions 2, 1 and 0 respectively identify the source register, as determined by the above Table 2-1, and 011 combination in bit positions 5, 4 and 3 respectively identify the destination register in the main set.

2-3.3 The Mnemonic Structure

1. The assembly language instructions for the Z-80 are given in convenient mnemonic forms for simple understanding and use. Basically, the instruction mnemonics

are structured in two parts as explained below:

(a) The Fixed Part - This is shown in capitals or upper case letters. The fixed part generally defines the function of the operation that the instruction performs. The letters in this part are arranged such that they convey the operation performed very easily. For example, INC is used for increment, DEC for decrement, SUB for subtraction and so on.

(b) The Variable Part - The variable part can consist of one or two operands or one or two addresses or displacements. The mnemonics for the variable part can be expressed by either the upper or the lower case letters. Generally, upper case letters are used to indicate the internal CPU registers and the lower case letters to identify other locations such as I/O devices or external memories. If two operands or displacements are involved in the variable part, then they are separated by a comma.

(c) Implied Operand - If the instructions are such that they contain implied operands, then the mnemonics for such instructions do not have variable parts.

2. If the operand is enclosed by parenthesis, it means that the contents of that location (usually a GP register) are used as a pointer to a memory location where the desired operand is stored or is to be stored.

3. The mnemonics are structured such that the fixed part of the instruction is followed by the destination (if a transfer is involved) which is followed by a comma. The source of the operand is the last item in the mnemonic structure.

4. Following the guidelines in paragraph 3 above, the assembly language instruction for the example mentioned in Section 2-3.2 would be written as follows:

$$\text{LD } r_1, r_2$$

where LD is the mnemonic for the load operation, r_1 represents the destination register (register E in this case) and r_2 represents the source (register D in this case). Of course when these mnemonics are translated into machine language by the assembler, they will appear as shown in Fig. 2-1.

5. Suppose we want to store the contents of accumulator A in a memory location whose address pointer is located in register pair BC. The assembly language instruction for this operation would be as follows:

$$\text{LD (BC), A}$$

2-3.4 Symbolic Expressions of Operations and Functions

All the instructions in the Z-80 set can be symbolically expressed. Before we describe and explain the actual instruction set, the symbology used is presented in Table 2-2.

Table 2-2 Instruction symbols.

SYMBOLS	FUNCTIONS
r	Denotes any register in the CPU GP main register set.
( )	Denotes contents of the register(s) are used as pointer to a memory location.
←	Denotes that the contents from the source are transferred to a destination. The tail of the arrow indicates the source and the head the destination.
n	Represents an 8-bit integer whose decimal value ranges from 0 to 255 which is commonly expressed as ⟨0, 255⟩
nn	Represents a 16-bit integer whose decimal value ranges from 0 to 65, 535 commonly shown as ⟨0, 65535⟩
dd	Represents any register pairs BC, DE, HL, SP.
qq	Represents any register pairs AF, BC, DE, HL.
$(PAIR)_H$	Represents the high-order byte of the register pair indicated in the parentheses, e.g., BC_H.
b	When used as subscript denotes the bit number (000 - 111) to be set, reset or tested in the indicated register.
$(PAIR)_L$	Represents the low-order byte of the register pair indicated in the parentheses, e.g., DE_H.
	Note: H and L when used as subscripts mean high and low order bytes of any register pair. They should not be confused with H and L registers in the CPU.

Table 2-2 (Continued)

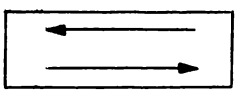
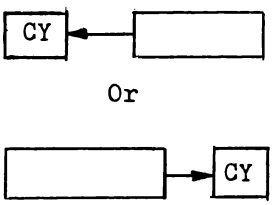
SYMBOLS	FUNCTIONS
⟷	A two-headed arrow indicates that the contents of the two registers are exchanged.
+	Means simple arithmetic add.
-	Means simple arithmetic subtract.
+ 1	Means increment (i.e. add 1).
- 1	Means decrement (i.e. subtract 1).
=	Means equal to.
≠	Means not equal to.
s	Any 8-bit location for all addressing modes allowed for that particular instruction.
ss	Any 16-bit location fro all addressing modes allowed for that particular instruction.
∧	Indicates logical AND operation.
∨	Indicates logical OR operation.
⊻	Indicates logical Exclusive-OR operation.
CY	Indicates Carry flip-flop.
$\overline{CY}$	Indicates complement Carry flip-flop.
	Indicates a register for shift or rotate operations. Arrow indicates the direction of shift or rotate.
	Indicates register shift or rotate with Carry flip-flop. Arrow indicates direction of shift or rotate.

Table 2-3 Status Flag Symbols.

SYMBOLS	EFFECT ON FLAGS
-	Flag unchanged by the operation.
*	Flag affected according ot the result of the operation.
1	Flag is set by the operation.
0	Flag is reset by the operation.
X	Flag is in a "don't care" status.
P	Flag is set according to parity of the result of the operation.
V	Flag is set according to the overflow resulting from the operation.
/	The content of the interrupt enable flip-flop (IFF) is copied into the P/V flag.
$	P/V flag is 0 if the result of BC - 1 = 0. Otherwise, P/V = 1.
¢	Z flag is 1 if A = (HL). Otherwise, Z = 0.
?	Flag is unknown.
@	If B - 1 = 0, Z flag is set. Otherwise it is reset.

2-3.5 The Status Register Symbols

Many instructions, when executed, give results whose status is stored in the appropriate flag or flip-flop in the Status Register F or F'. In our descriptions of the various instructions we include an indication of which status flags are affected, if any, and how they are affected. The symbols used to indicate the effect on each of the 6 status flags are given in Table 2-3.

2-3.6 The Register Pair Codes

In the Z-80 instructions which are given in tabular form in the rest of this chapter (as well as in chapters 3 and 4), the registers in the CPU that are involved in the particular instructions are identified in the OP codes by lower case letters(s). The binary codes that identify the register or the register pairs are given in the following tables.

Additionally, in the Bit Set, Rest and the Test group of instructions, the b code identifies the bit number that is involved. Table 2-10 gives the binary codes for the bits involved in the execution of these instructions.

Table 2-4 The r, r' Codes.

Register	r or r' Binary Codes
A	111
B	000
C	001
D	010
E	011
H	100
L	101

Table 2-5 The dd Codes.

Register Pair	qq Binary Codes
BC	00
DE	01
HL	10
AF	11

Table 2-6 The qq Codes.

Register Pair	dd Binary Codes
BC	00
DE	01
HL	10
SP	11

Table 2-7 The as Codes

Register Pair	ss Binary Codes
BC	00
DE	01
HL	10
SP	11

Table 2-8 The pp Codes

Register Pair	pp Binary Codes
BC	00
DE	01
IX	10
SP	11

Table 2-9 The rr Codes

Register Pair	rr Binary Codes
BC	00
DE	01
IY	10
SP	11

Table 2-10 Bit Indentification codes

Bit No.	b Binary Codes
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

2-4 THE INSTRUCTION SET

See Tables 2-11 through 2-17.

Table 2-11 8-bit Load Instructions Group

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS						OP CODE	NO. OF BYTES	OPERATION DESCRIPTION
			C	Z	P/V	S	N	H	76 543 210		
1.	LD r_1, r_2	$r_1 \leftarrow r_2$	-	-	-	-	-	-	01 r_1 r_2	1	Contents of r_2 loaded into r_1.
2.	LD r, n	r ← n	-	-	-	-	-	-	00 r 110 ← n →	2	8-bit integer is loaded into register r.
3.	LD r, (HL)	r ← (HL)	-	-	-	-	-	-	01 r 110	1	8-bit contents of memory location HL are loaded into register r.
4.	LD r, (IX + d)	r ← (IX + d)	-	-	-	-	-	-	11 011 101 01 r 110 ← d →	3	Contents of memory address, formed by sum of IX and displacement d, are loaded into register r.
5.	LD r, (IY + d)	r ← (IY + d)	-	-	-	-	-	-	11 111 101 01 r 110 ← d →	3	Contents of memory address, formed by sum of IY and displacement d, are loaded into register r.
6.	LD (HL), r	(HL) ← r	-	-	-	-	-	-	01 110 r	1	Contents of r are loaded into memory location specified by the HL register pair.
7.	LD (IX + d), r	(IX + d) ← r	-	-	-	-	-	-	11 011 101 01 110 r ← d →	3	Contents of r are loaded into memory location obtained by adding index register IX and displacement d.
8.	LD (IY + d), r	(IY + d) ← r	-	-	-	-	-	-	11 111 101 01 110 r ← d →	3	Contents of r are loaded into memory location obtained by adding index register IY and displacement d.

Table 2-11 (Continued) 8-bit Load Instructions Group.

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS						OP CODE	NO. OF BYTES	OPERATION DESCRIPTION
			C	Z	P/V	S	N	H	76 543 210		
9.	LD (HL), n	(HL) ← n	-	-	-	-	-	-	00 110 110 ← n →	2	Integer n is loaded into memory location specified by register pair HL.
10.	LD (IX + d), n	(IX + d) ← n	-	-	-	-	-	-	11 011 101 00 110 110 ← d → ← n →	4	Operand n is loaded into memory location specified by the sum of IX and the displacement d.
11.	LD (IY + d), n	(IY + d) ← n	-	-	-	-	-	-	11 111 101 00 110 110 ← d → ← n →	4	Operand n is loaded into memory location specified by the sum of IY and the displacement d.
12.	LD A, (BC)	A ← (BC)	-	-	-	-	-	-	00 001 011	1	Contents of memory location specified by the contents of BC registers are loaded into the accumulator.
13.	LD A, (DE)	A ← (DE)	-	-	-	-	-	-	00 011 010	1	Contents of memory location specified by the contents of DE registers are loaded into the accumulator.
14.	LD A, (nn)	A ← (nn)	-	-	-	-	-	-	00 111 010 ← n → ← n →	3	Contents of memory location specified by the contents of operands nn are loaded into the accumulator.

Table 2-11 (Continued) 8-bit Load Instructions Group.

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS						OP CODE	NO. OF BYTES	OPERATION DESCRIPTION
			C	Z	P/V	S	N	H	76 543 210		
15.	LD (BC), A	(BC) ← A	-	-	-	-	-	-	00 000 010	1	Contents of the accumulator are loaded into memory location specified by the contents of register pair BC.
16.	LD (DE), A	(DE) ← A	-	-	-	-	-	-	00 010 010	1	Contents of the accumulator are loaded into memory location specified by the contents of register pair DE.
17.	LD (nn), A	(nn) ← A	-	-	-	-	-	-	00 110 010 ← n → ← n →	3	Contents of the accumulator are loaded in memory location specified by the contents of operand nn.
18.	LD A, I	A ← I	-	*	/	-	-	-	11 101 101 01 010 111	2	Contents of Interrupt Vector Register I are loaded into the accumulator.
19.	LD A, R	A ← R	-	*	/	-	-	-	11 101 101 01 011 111	2	Contents of Memory Refresh Register R are loaded into the accumulator.
20.	LD I, A	I ← A	-	-	-	-	-	-	11 101 101 01 000 111	2	Contents of the accumulator are loaded into the Interrupt Vector Register I.
21.	LD R, A	R ← A	-	-	-	-	-	-	11 101 101 01 001 111	2	Contents of the accumulator are loaded into Memory Refresh Register R.

Table 2-15 Block Transfer and Search Instructions Group.

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS						OP CODE	NO. OF BYTES	OPERATION DESCRIPTION
			C	Z	P/V	S	N	H	76 543 210		
48.	LDI	(DE) ← (HL) DE ← DE + 1 HL ← HL + 1 BC ← BC - 1	-	-	* $	-	0	0	11 101 101 10 100 000	2	Contents of memory locations specified by HL are transferred to memory location specified by DE. Both HL and DE are incremented and BC (Byte Counter) is decremented.
49.	LDIR	(DE) ← (HL) DE ← DE + 1 HL ← HL + 1 BC ← BC - 1 Repeat until BC = 0.	-	-	0	-	0	0	11 101 101 10 110 000	2	The LDI instruction above is repeated. If BC = 0 then the instruction is terminated. If BC ≠ 0, then PC is decremented by 2 and the instruction is repeated.
50.	LDD	(DE) ← (HL) DE ← DE - 1 HL ← HL - 1 BC ← BC - 1	-	-	* $	-	0	0	11 101 101 10 101 000	2	Contents of memory locations specified by HL are transferred to memory location specified by DE. Then HL, DE and BC are decremented.
51.	LDDR	(DE) ← (HL) DE ← DE - 1 HL ← HL - 1 BC ← BC - 1 Repeat until BC = 0.	-	-	0	-	0	0	11 101 101 10 111 000	2	The LDD instruction above is repeated. If BC = 0, then the instruction is terminated. If BC ≠ 0, the PC is decremented by 2 and the instruction is repeated.
52.	CPI	A - (HL) HL ← HL + 1 BC ← BC - 1	-	* ¢	* $	*	*	*	11 101 101 10 100 001	2	Contents of memory location specified by HL are compared with the contents of the accumulator. If A = HL, then the Z flag is set HL incremented & BC decremented.

Table 2-11 (Continued) 8-bit Load Instructions Group.

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS						OP CODE	NO. OF BYTES	OPERATION DESCRIPTION
			C	Z	P/V	S	N	H	76 543 210		
15.	LD (BC), A	(BC) ← A	-	-	-	-	-	-	00 000 010	1	Contents of the accumulator are loaded into memory location specified by the contents of register pair BC.
16.	LD (DE), A	(DE) ← A	-	-	-	-	-	-	00 010 010	1	Contents of the accumulator are loaded into memory location specified by the contents of register pair DE.
17.	LD (nn), A	(nn) ← A	-	-	-	-	-	-	00 110 010 ← n → ← n →	3	Contents of the accumulator are loaded in memory location specified by the contents of operand nn.
18.	LD A, I	A ← I	-	*	/	-	-	-	11 101 101 01 010 111	2	Contents of Interrupt Vector Register I are loaded into the accumulator.
19.	LD A, R	A ← R	-	*	/	-	-	-	11 101 101 01 011 111	2	Contents of Memory Refresh Register R are loaded into the accumulator.
20.	LD I, A	I ← A	-	-	-	-	-	-	11 101 101 01 000 111	2	Contents of the accumulator are loaded into the Interrupt Vector Register I.
21.	LD R, A	R ← A	-	-	-	-	-	-	11 101 101 01 001 111	2	Contents of the accumulator are loaded into Memory Refresh Register R.

Table 2-12 16-bit Load Instructions Group.

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS						OP CODE	NO. OF BYTES	OPERATION DESCRIPTION
			C	Z	P/V	S	N	H	76 543 210		
22.	LD dd, nn	dd ← nn	-	-	-	-	-	-	00 dd0 001 ← n → ← n →	3	2-byte integer nn is locaded into dd register in the CPU.
23.	LD IX, nn	IX ← nn	-	-	-	-	-	-	11 011 101 00 100 001 ← n → ← n →	4	2-byte integer nn is loaded into the index register IX.
24.	LD IY, nn	IY ← nn	-	-	-	-	-	-	11 111 101 00 100 001 ← n → ← n →	4	2-byte integer nn is loaded into the index register IY.
25.	LD HL, (nn)	H ← (nn + 1) L ← (nn)	-	-	-	-	-	-	00 101 010 ← n → ← n →	3	Contents of memory location (nn) are loaded into low-order portion (i.e. L) and contents of next higher memory location (nn + 1) are loaded into high-order portion (i.e. H) of register pair HL.
26.	LD dd, (nn)	dd_H ← (nn + 1) dd_L ← (nn)	-	-	-	-	-	-	11 101 101 01 dd1 011 ← n → ← n →	4	Contents of memory location (nn) are loaded into the low-order position of dd and the contents of next higher location (nn + 1) are loaded in higher order position of dd.
27.	LD IX, (nn)	IX_H ← (nn + 1) IX_L ← (nn)	-	-	-	-	-	-	11 011 101 00 101 010 ← n → ← n →	4	Contents of memory location (nn) are loaded into low-order byte index register IX and contents of next higher location (nn + 1) are loaded into the high-byte of IX.

Table 2-12 (Continued) 16-bit Load Instructions Group.

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS C	Z	P/V	S	N	H	OP CODE 76 543 210	NO. OF BYTES	OPERATION DESCRIPTION
28.	LD IY, (nn)	$IY_H \leftarrow (nn + 1)$ $IY_L \leftarrow (nn)$	-	-	-	-	-	-	11 111 101 00 101 010 ← n → ← n →	4	Contents of memory location (nn) are loaded into low-order byte of index register IY and contents of higher location (nn + 1) are loaded into high-order byte of IY.
29.	LD (nn), HL	$(nn + 1) \leftarrow H$ $(nn) \leftarrow L$	-	-	-	-	-	-	00 100 010 ← n → ← n →	3	Contents of low-byte of pair HL are loaded into memory location (nn) and contents of high-byte of H are loaded into next location (nn + 1).
30.	LD (nn), dd	$(nn + 1) \leftarrow dd_H$ $(nn) \leftarrow dd_L$	-	-	-	-	-	-	11 101 101 01 dd0 011 ← n → ← n →	4	Low-byte of register pair dd is loaded into memory location (nn) high-byte of dd is loaded into (nn + 1).
31.	LD (nn), IX	$(nn + 1) \leftarrow IX_H$ $(nn) \leftarrow IX_L$	-	-	-	-	-	-	11 011 101 00 100 010 ← n → ← n →	4	Low byte of index register IX is loaded into memory location (nn) and high-byte of IX is loaded into (nn + 1).
32.	LD (nn), IY	$(nn+ 1) \leftarrow IY_H$ $(nn) \leftarrow IY_L$	-	-	-	-	-	-	11 111 101 00 100 010 ← n → ← n →	4	Low-byte of index register IY is loaded into memory location (nn) and high-byte of IY is loaded into (nn + 1).

Table 2-12 (Continued) 16-bit Load Instructions Group.

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS						OP CODE	NO. OF BYTES	OPERATION DESCRIPTION
			C	Z	P/V	S	N	H	76 543 210		
33.	LD SP, HL	SP ← HL	-	-	-	-	-	-	11 111 001	1	Contents of pair HL are loaded into the Stack Pointer SP.
34.	LD SP, IX	SP ← IX	-	-	-	-	-	-	11 011 101 11 111 001	2	Contents of index register IX are loaded into Stack Pointer SP.
35.	LD SP, IY	SP ← IY	-	-	-	-	-	-	11 111 101 11 111 001	2	Contents of index register IY are loaded into the Stack Pointer SP.

Table 2-13 Stack Push/Pop, 16-bit, Instructions Group

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS						OP CODE	NO. OF BYTES	OPERATION DESCRIPTION
			C	Z	P/V	S	N	H	76 543 210		
36.	PUSH qq	$(SP - 1) \leftarrow qq_H$ $(SP - 2) \leftarrow qq_L$	-	-	-	-	-	-	11 qq0 101	1	Stack Pointer is first decremented and high-byte of register qq is pushed into this location. SP is again decremented and low-byte of qq is pushed into this location.
37.	PUSH IX	$(SP - 1) \leftarrow IX_H$ $(SP - 2) \leftarrow IX_L$	-	-	-	-	-	-	11 011 101 11 100 101	2	Stack Pointer is decremented and high-byte of IX pair is pushed into this location. SP is decremented again and low-byte of IX is pushed into this location.
38.	PUSH IY	$(SP - 1) \leftarrow IY_H$ $(SP - 2) \leftarrow IY_L$	-	-	-	-	-	-	11 111 101 11 100 101	2	Stack Pointer is decremented and high-byte of IY pair is pushed into this location. SP is decremented again and low-byte of IY is pushed into this location.
39.	POP qq	$qq_L \leftarrow (SP)$ $qq_H \leftarrow (SP + 1)$	-	-	-	-	-	-	11 qq0 001	1	Contents of Stack Pointer (low-byte) are loaded into register pair qq low-byte. The SP is then incremented and contents of this location are loaded in qq high-byte.

Table 2-13 (Continued) Stack Push/Pop, 16-bit, Instructions Group

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS						OP CODE	NO. OF BYTES	OPERATION DESCRIPTION
			C	Z	P/V	S	N	H	76 543 210		
40.	POP IX	$IX_L \leftarrow (SP)$ $IX_H \leftarrow (SP + 1)$	-	-	-	-	-	-	11 011 101 11 000 001	2	Contents of SP are loaded in low-byte of IX. SP is then incremented and contents of this location are loaded in the high-byte of IX.
41.	POP IY	$IY_L \leftarrow (SP)$ $IY_H \leftarrow (SP + 1)$	-	-	-	-	-	-	11 111 101 11 100 001	2	Contents of SP are loaded in low-byte of IY. SP is then incremented and contents of this location are loaded in the high-byte of IY.

Table 2-14 Exchange Instructions Group

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS						OP CODE	NO. OF BYTES	OPERATION DESCRIPTION
			C	Z	P/V	S	N	H	76 543 210		
42.	EX DE, HL	DE ⟷ HL	-	-	-	-	-	-	11 101 011	1	Contents of register pairs DE and HL are exchanged.
43.	EX AF, A'F'	AF ⟷ A'F'	-	-	-	-	-	-	00 001 000	1	Contents of register pairs AF and A'F' are exchanged.
44.	EXX	BC ⟷ B'C' DE ⟷ D'E' HL ⟷ H'L'	-	-	-	-	-	-	11 011 001	1	Contents of BC and B'C' are exchanged, contents of DE and D'E' are exchanged, and cintents of HL and H'L' are exchanged.
45.	EX (SP), HL	L ⟷ (SP) H ⟷ (SP + 1)	-	-	-	-	-	-	11 100 011	1	Low-byte of the HL pair is exchanged with the location specified by SP. SP is then incremented and the high-byte of HL is exchanged with the memory location specified by SP + 1.
46.	EX (SP), IX	IX_L ⟷ (SP) IX_H ⟷ (SP + 1)	-	-	-	-	-	-	11 011 101 11 100 011	2	Low-byte of index register IX is exchanged with the contents of memory location specified by SP. SP is then incremented. High-byte of IX is then exchanged with the location specified by SP + 1.
47.	EX (SP), IY	IY_L ⟷ (SP) IY_H ⟷ (SP + 1)	-	-	-	-	-	-	11 111 101 11 100 011	2	Low-byte of index register IY is exchanged with the contents of memory location specified by SP. SP is then incremented. High-byte of IY is then exchanged with the location specified by SP + 1.

Table 2-15 Block Transfer and Search Instructions Group.

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS						OP CODE	NO. OF BYTES	OPERATION DESCRIPTION
			C	Z	P/V	S	N	H	76 543 210		
48.	LDI	(DE) ← (HL) DE ← DE + 1 HL ← HL + 1 BC ← BC - 1	-	-	* $	-	0	0	11 101 101 10 100 000	2	Contents of memory locations specified by HL are transferred to memory location specified by DE. Both HL and DE are incremented and BC (Byte Counter) is decremented.
49.	LDIR	(DE) ← (HL) DE ← DE + 1 HL ← HL + 1 BC ← BC - 1 Repeat until BC = 0.	-	-	0	-	0	0	11 101 101 10 110 000	2	The LDI instruction above is repeated. If BC = 0 then the instruction is terminated. If BC ≠ 0, then PC is decremented by 2 and the instruction is repeated.
50.	LDD	(DE) ← (HL) DE ← DE - 1 HL ← HL - 1 BC ← BC - 1	-	-	* $	-	0	0	11 101 101 10 101 000	2	Contents of memory locations specified by HL are transferred to memory location specified by DE. Then HL, DE and BC are decremented.
51.	LDDR	(DE) ← (HL) DE ← DE - 1 HL ← HL - 1 BC ← BC - 1 Repeat until BC = 0.	-	-	0	-	0	0	11 101 101 10 111 000	2	The LDD instruction above is repeated. If BC = 0, then the instruction is terminated. If BC ≠ 0, the PC is decremented by 2 and the instruction is repeated.
52.	CPI	A - (HL) HL ← HL + 1 BC ← BC - 1	-	* ¢	* $	*	*	*	11 101 101 10 100 001	2	Contents of memory location specified by HL are compared with the contents of the accumulator. If A = HL, then the Z flag is set HL incremented & BC decremented.

Table 2-15 (Continued) Block Transfer and Search Instructions Group.

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS						OP CODE	NO. OF BYTES	OPERATION DESCRIPTION
			C	Z	P/V	S	N	H	76 543 210		
53.	CPIR	A - (HL) HL ← HL + 1 BC ← BC - 1 Repeat until A = (HL) or B = 0.	-	*	*	*	*	*	11 101 101 10 110 001	2	The CPI instruction above is repeated. If A = (HL), i.e. a match is found, the instruction is terminated. If BC≠ 0, then PC is decremented by 2 and the instruction is repeated.
				¢	$						
54.	CPD	A - (HL) HL ← HL - 1 BC ← BC - 1	-	*	*	*	*	*	11 101 101 10 101 001	2	Contents of memory location specified by HL are compared with the contents of the accumulator. If A = HL, the Z flag is set. Then both HL and BC are decremented.
				¢	$						
55.	CPDR	A - (HL) HL ← HL - 1 BC ← BC - 1 Repeat until A = (HL) or BC = 0.	-	*	*	*	*	*	11 101 101 10 111 001	2	The CPD instruction above is repeated. If A = HL, i.e. a match is found, the instruction is terminated. If both BC ≠ 0, then the PC is decremented by 2 and the instruction is repeated.
				¢	$						

Table 2-16 8-bit Arithmetic and Logic Instructions Group.

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS						OP CODE	NO. OF BYTES	OPERATION DESCRIPTION
			C	Z	P/V	S	N	H	76 543 210		
56.	ADD A, r	A ← A + r	*	*	V	*	0	*	10 000 r	1	Contents of accumulator and register r are added and the result is stored in the accumulator.
57.	ADD A, n	A ← A + n	*	*	V	*	0	*	11 000 110 ← n →	2	Integer n is added to the accumulator and the result is stored in the accumulator.
58.	ADD A, (HL)	A ← A + (HL)	*	*	V	*	0	*	10 000 110	1	Contents of memory location specified by the contents of HL are added to the accumulator and the result is stored in the accumulator.
59.	ADD A, (IX + d)	A ← A + (IX + d)	*	*	V	*	0	*	11 011 101 10 000 110 ← d →	3	Addition of displacement d and the contents of index register IX point to memory location whose contents are added to the accumulator. Result is stored in the accumulator.
60.	ADD A, (IY + d)	A ← A + (IY + d)	*	*	V	*	0	*	11 111 101 10 000 110 ← d →	3	Addition of displacement d and the contents of index register IY point to memory location whose contents are added to the accumulator. Result is stored in the accumulator.

Table 2-16 (Continued) 8-bit Arithmetic and Logic Instructions Group.

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS						OP CODE	NO. OF BYTES	OPERATION DESCRIPTION
			C	Z	P/V	S	N	H	76 543 210		
61.	ADC A, s	A ← A + s + CY	*	*	V	*	0	*	001	-	s operand includes r, n, (HL), (IX + d) and (IY + d). s is added to the accumulator with carry and result is stored in the accumulator. (Note: Bits 5,4 and 3, as shown here are substituted for corresponding bits in the OP codes of instructions # 56-60 for corresponding operations. In instruction #57 it is substituted in byte #1 but in instructions 59 and 60 it is substituted in byte # 2.)
											The student should note that the explanation in instruction #61 above regarding bits 5, 4 and 3 in the OP codes also applies to instructions #62-67.
62.	SUB s	A ← A - s	*	*	V	*	1	*	010	-	Operand s (same as instruction #61) is subtracted from the accumulator and the result is stored in the accumulator.
63.	SBC A, s	A ← A - s - CY	*	*	V	*	1	*	011	-	Operand s (same as instruction #61) and the carry are subtracted from the accumulator and the result is stored in the accumulator.

Note: For instructions 61-63, construct the OP code as shown in instructions 56-60 but with appropriate changes in bits 5, 4 and 3 as shown above. For index registers changes are only made in the second byte.

Table 2-16 (Continued) 8-bit Arithmetic and Logic Instructions Group.

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS						OP CODE	NO. OF BYTES	OPERATION DESCRIPTION
			C	Z	P/V	S	N	H	76 543 210		
64.	AND s	A ← A ∧ s	0	*	P	*	0	1	100	-	Operand s (same as instruction #61) is AND-ed with the accumulator and the result is stored in the accumulator.
65.	OR s	A ← A ∨ s	0	*	P	*	0	0	110	-	Operands s (same as instruction #61) is OR-ed with the accumulator and the result is stored in the accumulator.
66.	XOR s	A ← A ⊻ s	0	*	P	*	0	0	101	-	Operand s (same as instruction #61) is EXCLUSIVE OR-ed with the accumulator and the result is stored in the accumulator.
67.	CP s	A - s	*	*	V	*	1	*	111	-	Operand s (same as instruction #61) is compared with the accumulator. If A = s, then the Z flag is set. The accumulator contents are not altered.
68.	INC r	r ← r + 1	-	*	V	*	0	*	00 r 100	1	Contents of r are incremented.
69.	INC (HL)	(HL) ← (HL) + 1	-	*	V	*	0	*	00 110 100	1	Contents of memory location addressed by HL are incremented.

Note: For instructions 64-67, construct the OP code as shown in instructions 56-60, but with appropriate changes in bits 5, 4 and 3 as shown above. For index register make changes only in the second byte.

Table 2-16 (Continued) 8-bit Arithmetic and Logic Instructions Group.

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS						OP CODE	NO. OF BYTES	OPERATION DESCRIPTION
			C	Z	P/V	S	N	H	76 543 210		
70.	INC (IX + d)	(IX + d) ← (IX + d)+1	-	*	V	*	0	*	11 011 101 00 110 100 ← d →	3	Displacement d is added to the contents of the index register IX. The contents of the memory addressed by this sum are incremented.
71.	INC (IY + d)	(IY + d) ← (IY + d) + 1	-	*	V	*	0	*	11 111 101 00 110 100 ← d →	3	Displacement d is added to the contents of the index register IY. The contents of the memory addressed by this sum are incremented.
72.	DEC m	m ← m - 1	-	*	V	*	1	*	101	-	Operand m includes r, (HL). (IX + d) and (IY + d). The contents of operand m are decremented. (Note: Bits 2, 1 and 0, as shown here, are substituted for corresponding bits in the OP codes of instructions #68-71 for corresponding operations. In instructions #68-69, it is substituted in byte #1, but in instructions #70-71, it is substituted in byte #2.

Table 2-17 16-bit Arithmetic Instructions Group.

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS						OP CODE	NO. OF BYTES	OPERATION DESCRIPTION
			C	Z	P/V	S	N	H	76 543 210		
73.	ADD HL, ss	HL ← HL + ss	*	-	-	-	0	?	00 ss1 001	1	Contents of register pair ss are added to HL and the result is stored in HL.
74.	ADC HL, ss	HL ← HL + ss + CY	*	*	V	*	0	?	11 101 101 01 ss1 010	2	Contents of register pair ss and Carry are added to HL and the result is stored in HL.
75.	SBC HL, ss	HL ← HL - ss - CY	*	*	V	*	0	?	11 101 101 01 ss0 010	2	Contents of pair ss and Carry are subtracted from HL and the result is stored in HL.
76.	ADD IX, pp	IX ← IX + pp	*	-	-	-	0	?	11 011 101 00 pp1 001	2	Contents of pair pp are added to the index register IX and the result is stored in IX.
77.	ADD IY, rr	IY ← IY + rr	*	-	-	-	0	?	11 111 101 00 rr1 001	2	Contents of pair rr are added to the index register IY and the result is stored in IY.
78.	INC ss	ss ← ss + 1	-	-	-	-	-	-	00 ss0 011	1	Contents of register pair ss are incremented.
79.	INC IX	IX ← IX + 1	-	-	-	-	-	-	11 011 101 00 100 011	2	Contents of index register IX are incremented.
80.	INC IY	IY ← IY + 1	-	-	-	-	-	-	11 111 101 00 100 011	2	Contents of index register IY are incremented.

Table 2-17 (Continued) 16-bit Arithmetic Instructions Group.

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS						OP CODE	NO. OF BYTES	OPERATION DESCRIPTION
			C	Z	P/V	S	N	H	76 543 210		
81.	DEC ss	ss ← ss - 1	-	-	-	-	-	-	00 ss1 011	1	Contents of pair ss are decremented.
82.	DEC IX	IX ← IX - 1	-	-	-	-	-	-	11 011 101 00 101 011	2	Contents of index register IX are decremented.
83.	DEC IY	IY ← IY - 1	-	-	-	-	-	-	11 111 101 00 101 011	2	Contents of index register IY are decremented.

2-5 REVIEW QUESTIONS

2-1 Indicate true or false for the Load and Exchange instructions group.

a. ___ Involves both 8-bit and 16-bit data transfers.

b. ___ Involves data transfers between internal registers of the CPU.

c. ___ Does not involve data transfers between the CPU and the external memory.

d. ___ The transfers destroy the data in the source locations.

e. ___ Instructions specify both the source and destination.

2-2 The load instructions include _____ instructions which allow the operand contained in the instruction to be transferred. (load immediate; implied mode; indexed mode; relative mode)

2-3 The following statements relate to the Block Transfer and Search instructions group. Indicate true or false.

a. ___ Block of data can be transferred from one memory location to another.

b. ___ Block transfers require as many instructions as there are bytes in the block.

c. ___ An entire block of data of any length can be searched by just one instruction.

d. ___ When the desired word is found, the search is automatically terminated.

e. ___ If the desired word is not found, the search is continued in the next block.

2-4 Operands for arithmetic and logic operations are located in (circle all that apply):

a. The accumulator in the CPU.
b. The GP registers of the CPU.
c. The external I/O devices.
d. The external memory.

2-5 The results of the arithmetic/logic operations are placed in the _____. (Index register; GP registers; accumulator; instruction register)

2-6 The arithmetic/logic instructions of the Z-80 are capable of performing (circle all that apply):

a. 8-bit subtractions.
b. 16-bit logical operations.
c. 16-bit additions.
d. 8-bit additions.

2-7 Indicate true or false for the Bit Manipulation group of instructions.

a. ___ Any bit in the accumulator can be set.

b. ___ Any bit in the CPU registers can be set or reset.

c. ___ Bits in the external memory can be set and reset.

d. ___ Bits in the external memory can be tested but not set or reset.

2-8 The following statements apply to the I/O group of instructions. Indicate true or false.

a. ___ Data can be transferred between the I/O devices and the external memory.

b. ___ A single instruction can transfer a block of data of a maximum of 512 bytes.

c. ___ Parity errors in serial transmissions cannot be detected.

d. ___ Additional operations (i.e. Instructions) are needed to determine the status of the incoming data.

2-9 From the following statements pick out those that apply to the generalized instruction formats of the Z-80.

a. The instruction could consist of 1,2, 3 or 4 bytes.
b. If the OP code contains 2 bytes, then the following byte must always be an operand.
c. If the OP code contains 2 bytes, then the following byte must be an address byte.
d. In a 2-byte operand, the first byte is the high-order byte.
e. In a 2-byte address, the second byte is always the high-order byte.
f. The second byte of a 2-byte instruction could be either an operand or an address.

2-10 The contents of register E in the main set are to be loaded into register C of the same register set. Circle the correct OP code for this operation.

a. 00 001 011
b. 01 011 001
c. 11 000 101
d. 01 001 011
e. 10 011 001
f. 10 001 011

2-11 Refer to question 2-10 above. The correct assembly language mnemonic for this instruction is,

a. LD E, C
b. LD C, E
c. LD r_2, r_1
d. LD r_1, r_2
e. LD p, p
f. LD q, q

2-12 The following statements relate to the Z-80 mnemonic structure. Indicate true or false.

a. ___ The fixed part of the mnemonic defines the function to be performed.

b. ___ The fixed part is shown in capitals.

c. ___ The variable part is generally shown in lower case letters.

d. ___ Two operands or displacements in the variable part are separated by a comma.

e. ___ If the instruction contains an implied operand, then the mnemonic does not have a variable part.

2-13 The lower case letter n represents _____ bit integer whose decimal value ranges from 0 to _____ . (8; 16; 255; 256; 65,535; 65,536)

2-14 Any CPU register pairs BC, DE, HL, SP are represented by _____ . (rr'; pp; nn; dd; qq)

2-15 The following statements relate to the mnemonic LD r, (HL). Indicate true or false.

a. ___ Contents of HL are transferred to any GP register r.

b. ___ Contents of HL are a memory address.

c. ___ Contents of register r are transferred into HL pair.

d. ___ Contents of memory addressed by HL are transferred to CPU register r.

2-16 We wish to load an integer n into the memory location whose address is obtained by adding a displacement to the contents of the index register IX. Pick out the proper symbolic representation for this operation.

a. $n \leftarrow IX + d$
b. $IX + d \leftarrow n$
c. $(IX + d) \leftarrow n$
d. $d + IX \leftarrow n$

2-17 The contents of the index register IY are pushed into the Stack. The correct symbolic representation for this operation is:

a. $(SP - 1) \leftarrow IY_H$
   $(SP - 2) \leftarrow IY_L$

b. $(SP - 1) \leftarrow IY_L$
   $(SP - 2) \leftarrow IY_H$

c. $SP \leftarrow IY$

d. $IY \leftarrow SP$

2-18 We want to exchange the contents of HL with the contents of HL', The correct mnemonic for this instruction is,

a. LD HL, HL'
b. LD HL', HL
c. EXX
d. EX HL', HL
e. EX HL, HL'

2-19 We want to transfer the contents of HL to DE and then decrement both DE and HL. What are the correct mnemonics for this?

a. LDD
b. LDH
c. LDHD
d. LDDH

2-20 Write down the symbolic representation for the compare mnemonic CPD.

2-21 We want to add (IX + d) to the accumulator and store the result in the accumulator. The correct mnemonic for this is:

a. LD A, (IX + d)
b. ADD A, (IX + d)
c. LD (IX + d), A
d. ADD (IX + d), A

2-22 The symbolic representation $A \leftarrow A \vee S$ means that

a. Operand s is OR-ed with the accumulator.
b. The result of the OR operation is stored in the accumulator.
c. s is AND-ed with the accumulator.

3 THE Z-80 INSTRUCTION SET (PART TWO)

CHAPTER OBJECTIVES

The objectives of this chapter are as follows:

1. To present the ROTATE and SHIFT group of instructions.

2. To present the BIT MANIPULATION group of instructions which include the Set, Reset and Test operations.

3. To present the JUMP group of instructions which includes both conditional and and unconditional JUMPS.

4. To present the CALL and RETURN group of instructions which includes both conditional and unconditional BRANCHES

5. To explain arithmetic and logic shifts (right and left) as used in the Z-80/CPU.

TEXTBOOK REFERENCES

For reviewing material in the main hardcover textbook, relevant to the topics covered in this chapter, the following chapters and/or sections are suggested.

1. For ROTATE and SHIFT instructions - Sec. 4-2.5
2. For JUMP instructions and operations - Sec. 4-2.3.2
3. For CALL and RETURN instructions and operations - Sec. 4-2.3.3

3-1 INTRODUCTION

This chapter is a continuation of the Z-80 instruction set which was introduced in Chapter 2. Four groups of instructions are covered in this chapter. The rest of the instructions are covered in Chapter 4. The instructions are presented in the same convenient tabular format that was used in Chapter 2. All the status flag symbols that are presented in Table 2-3 also apply here. Likewise, the instruction symbols presented in Table 2-2 also apply to the instructions in this chapter.

3-2 SHIFT OPERATIONS

3-2.1 Arithmetic Shifts

3-2.1.1 Shift Left Arithmetic (SLA)

In any computer, including μCs, operands involved in arithmetic operations must include a proper sign bit to identify the operand as either a positive or a negative quantity. The most-significant-bit (MSB) of the word is the sign bit. The convention for the sign bit is:

0 - indicates a positive quantity
1 - indicates a negative quantity

Arithmetic operations, by their very nature, require that the sign bit be preserved regardless of the shifting process. In the Z-80 system when a left shift arithmetic is performed, either on a CPU register or on a word in the memory, the MSB of the word is shifted and saved in the Carry flag, i.e. the CY flip-flop. During such a shift, a 0 is automatically inserted in the vacated LSB position of the word. This operation is graphically shown in Fig. 3-01 which includes the contents of the register before and after the shift.

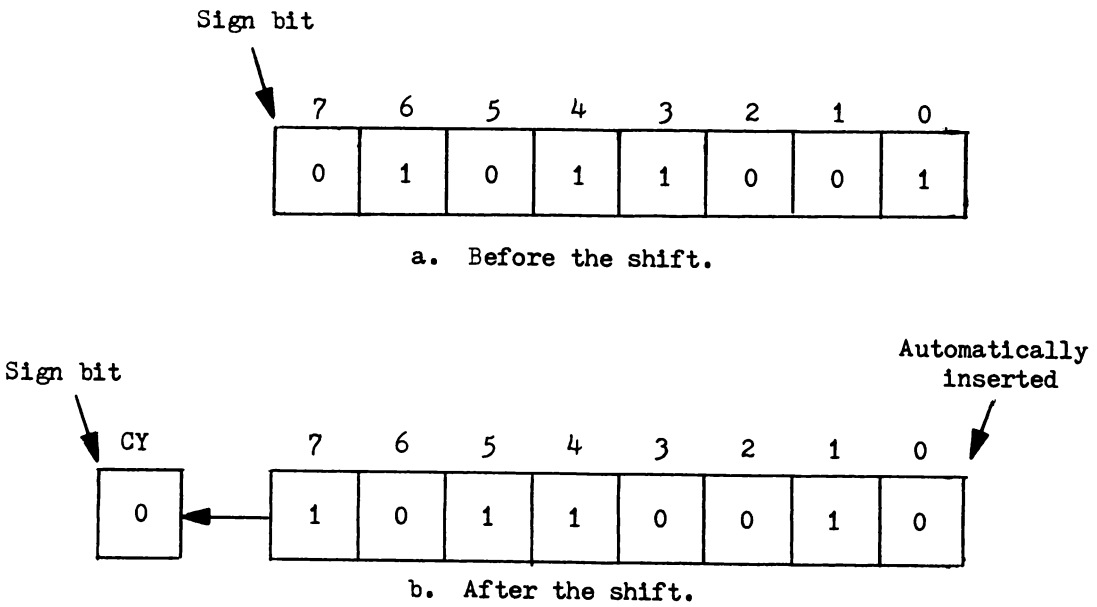


Fig. 3-01 Shift left arithmetic (SLA).

3-2.1.2 Shift Right Arithmetic (SRA)

In this operation, the sign bit is saved in the original bit 7 position and replicated in the next lower position of the register, i.e. bit 6 position. The LSB of the register (i.e. bit 0 position) is transferred into the CY flip-flop and saved there. Figure 3-02 shows the contents of the register (or memory location) before and after the shifting operation.

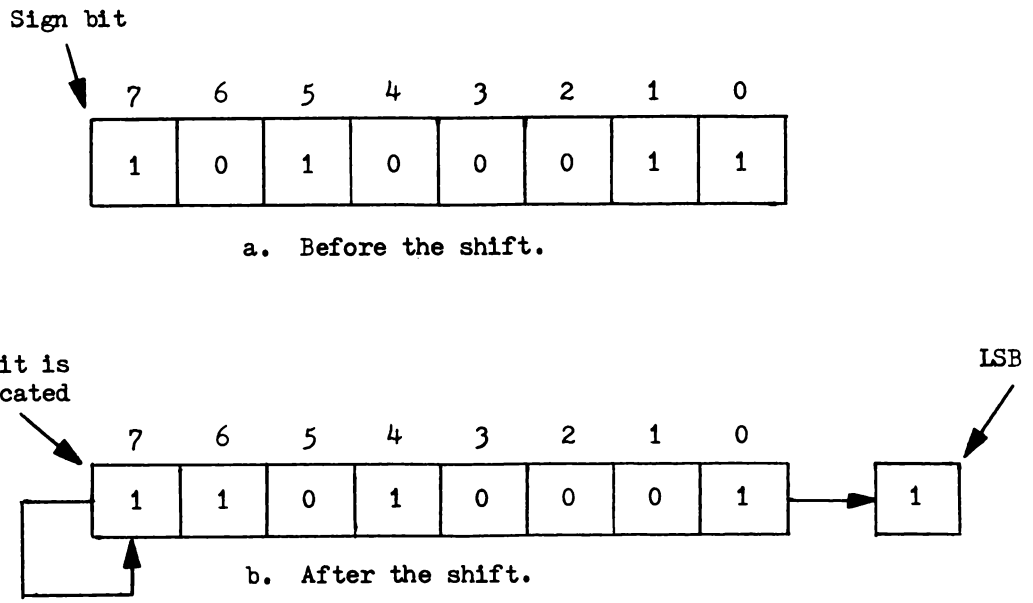
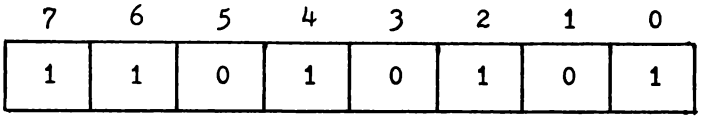


Fig. 3-02 Shift right arithmetic (SRA)

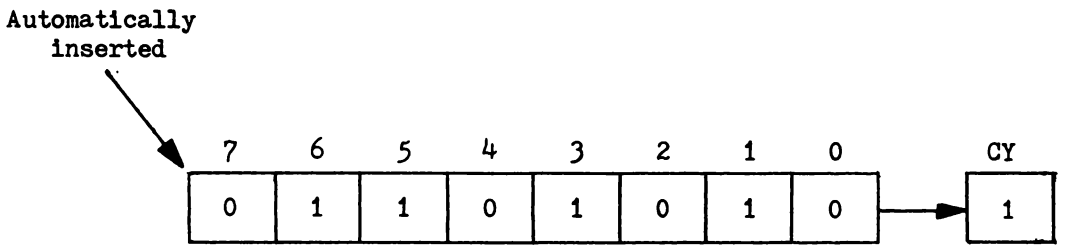
3-2.1.3 Shift Right Logic (SRL)

A word or an operand that is to be used in an arithmetic operation must have a sign bit associated with it. However, in a word to be used for a logical operation the sign bit is meaningless and so is not used. In a shift right logic operation in the Z-80 system, a 0 is automatically inserted in the vacated MSB position, i.e. the bit 7 position, of the register. The LSB is shifted into the CY flip-flop and saved there. A typical example of this operation, before and after the shift, is shown in Fig. 3-03.

Note: In the Z-80 system there is no instruction for shift left logic.



a. Before the shift.



b. After the shift.

Fig. 3-03 Shift right logic (SRL)

3-2.2 Register/Memory Symbols for Rotate/Shift

In describing the Rotate and Shift instructions, descriptive figures for the various registers and memory locations, involved in these operations and the Carry flip-flop, are used. It is inconvenient to include these figures in the symbolic operations columns of the instructions in the tables that follow. We get around this problem by first giving all the figures in this section and assigning a figure number to each. We will then insert only the figure number in the appropriate column of the tables and we will include the identity of the register involved in the operation, such as A for the accumulator. In these tables note the following.

r - denotes any register in the CPU main register set.
m - denotes r, (HL), (IX + d) and (IY + d).
A - denotes the accumulator.

3-2.3 The Mnemonic Convention

The following explanation is intended to help students with the proper inerpretation of the mnemonics used in Table 3-1.

1. In instruction 84 to 94, the first letter, R, means rotate or recirculate.

2. The second letter, L, means left and the second letter, R, means right.

Fig. 3-1

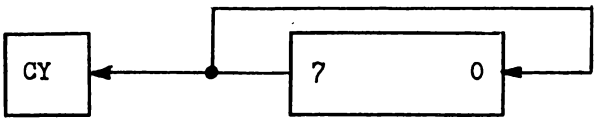


Fig. 3-2

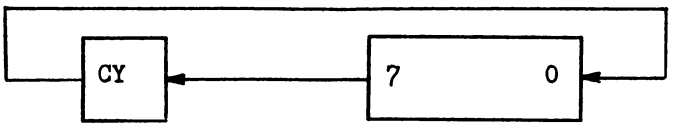


Fig. 3-3

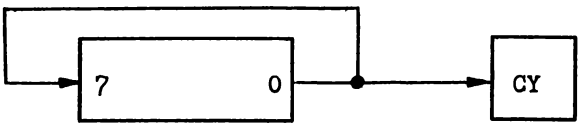


Fig. 3-4

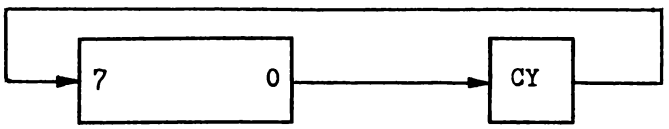


Fig. 3-5

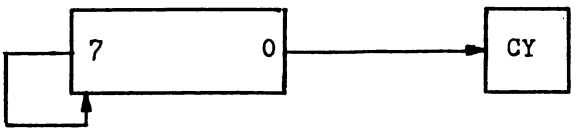


Fig. 3-6

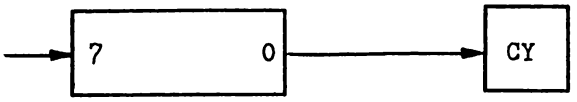


Fig. 3-7

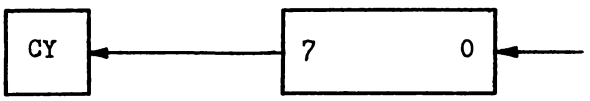


Fig. 3-8

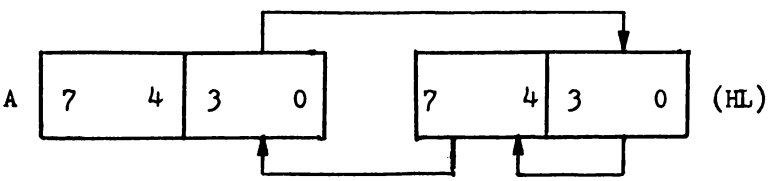
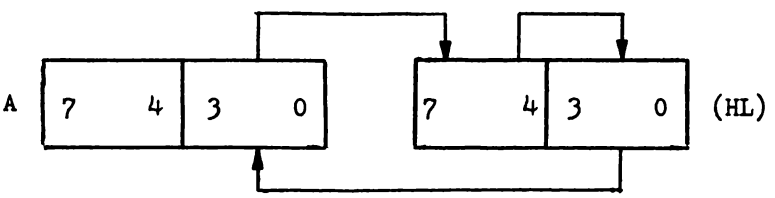


Fig. 3-9



Symbols for SHIFT/ROTATE operations.

3. The third letter, C, whenever it appears, means that the rotation takes place on the contents of the register (or the memory location) only. However, the end bit of the word, either the LSB or the MSB, is shifted into the CY flip-flop.

4. The third or the fourth letter, A, refers to the accumulator while letters r and m refer to the registers or memory locations which were defined in Sec. 3-2.2.

5. For instructions 95, 96 and 97, the interpretation of the mnemonics for the three capital letters is as follows:

    a. SLA means Shift Left Arithmetic.
    b. SRA means Shift Right Arithmetic
    c. SRL means Shift Right Logic.

6. For instructions 98 and 99 operation descriptions, given in Table 3-1, give the proper interpretations.

3-2.4 OP Code Construction

In Table 3-1 notice that, for instructions 92 to 97, only three bits are given for the OP code. These three bit codes are intended to be substituted for the corresponding three bits (bits 5, 4 and 3) in the last byte of the related RLC instructions, i.e. instructions 88 to 91. The rest of the OP code is the same. The following examples clarify how this is done.

Example 3-1

Write down the complete OP code for the RRC m instruction (instruction 93) for the case where the operand is stored in a location whose address is contained in the HL register pair of the CPU.

Solution

The final mnemonics for this instruction is

RRC (HL)

To get the OP code, we go to RLC (HL) whose OP code (from instruction 89) is

Byte 1	11 001 011
Byte 2	00 000 110

To get the appropriate OP code for RRC (HL), we get the three bits 001 from instruction 93 in Table 3-1 and substitute them for bits 5, 4 and 3 in Byte 2 above.

Thus, the correct OP code for RRC (HL) is

Byte 1	11 001 011
Byte 2	00 001 110

Example 3-2

Write down the complete OP code for the RR m instruction (instruction 94) for the situation where the operand is stored in a memory location whose address is obtained by adding the contents of the Index Register Y to the displacement in the instruction.

Solution

The final mnemonic for this instruction is RR (IY + d). To get the OP code, we go to RLC (IY + d) (instruction 91) whose OP code is

Byte 1	11 111 101
Byte 2	11 001 011
Byte 3	← d →
Byte 4	00 000 110

To get the appropriate OP code for RR (IY + d), we get the three bits 011 from instruction 94 in Table 3-1 and substitute them for bits 5, 4 and 3 in byte 4 above. Thus the correct OP code for RR (IY + d) is

Byte 1	11 111 101
Byte 2	11 001 011
Byte 3	← d →
Byte 4	00 011 110

3-3 ROTATE AND SHIFT GROUP

Table 3-1 Rotate and Shift Instructions Group.

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS						OP CODE	NO. OF BYTES	OPERATION DESCRIPTION
			C	Z	P/V	S	N	H	76 543 210		
84	RLCA	Fig. 3-1 A	*	-	-	-	0	0	00 000 111	1	Contents of accumulator are rotated left one bit. Sign bit is also transferred in CY flag.
85	RLA	Fig. 3-2 A	*	-	-	-	0	0	00 010 111	1	Contents of accumulator, includig the CY flag, are rotated left one bit.
86	RRCA	Fig. 3-3 A	*	-	-	-	0	0	00 001 111	1	Contents of accumulator are rotated right one bit. Bit 0 is transferred into the CY flag.
87	RRA	Fig. 3-4 A	*	-	-	-	0	0	00 011 111	1	Contents of accumulator, including CY flag, are rotated right one bit.
88	RLC r	Fig. 3-1 r	*	*	P	*	0	0	11 001 011 00 000 r	2	Contents of register r are rotated left one bit. Bit 7 is also transferred to the CY flag.
89	RLC (HL)	Fig. 3-1 (HL)	*	*	P	*	0	0	11 001 011 00 000 110	2	Contents of memory location addressed by HL are rotated left one bit. Bit 7 is also transferred into the CY flag.
90	RLC (IX + d)	Fig. 3-1 (IX + d)	*	*	P	*	0	0	11 011 101 11 001 011 ← d → 00 000 110	4	Contents of memory location specified by sum of index register IX and displacement d are rotated left one bit. Bit 7 is also transferred into the CY flag.

Table 3-1 (Continued) Rotate and Shift Instructions Group

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS						OP CODE	NO. OF BYTES	OPERATION DESCRIPTION
			C	Z	P/V	S	N	H	76 543 210		
91	RLC (IY + d)	Fig. 3-1 (IY + d)	*	*	P	*	0	0	11 111 101 11 001 011 ← d → 00 000 110	4	Contents of memory location specified by sum of index register IY and displacement d are rotated left one bit. Bit 7 is also transferred to CY flag.
92	RL m	Fig. 3-2 m	*	*	P	*	0	0	010	-	The m operand (as defined for RLC codes above) is rotated left one bit including the CY flag.
93	RRC m	Fig. 3-3 m	*	*	P	*	0	0	001	-	The m operand (as defined for RLC codes above) is rotated right one bit. Bit 0 is also transferred to the CY flag.
94	RR m	Fig. 3-4 m	*	*	P	*	0	0	011	-	The operand m (as defined for RLC codes above) is rotated right one bit including the CY flag.
95	SLA m	Fig. 3-7 m	*	*	P	*	0	0	100	-	The m operand (as defined for RLC codes above) is shifted left one bit including the CY flag.
96	SRA m	Fig. 3-5 m	*	*	P	*	0	0	101	-	Operand m (as defined for RLC codes above) is right shifted one bit. LSB is shifted into CY and MSB is replicated.
97	SRL m	Fig. 3-6 m	*	*	P	*	0	0	111	-	Operand m (as defined for RLC codes above) is shifted right one bit. Bit 0 is shifted into the CY flag.

Table 3-1 (Continued) Rotate and Shift Instructions Group.

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS						OP CODE	NO. OF BYTES	OPERATION DESCRIPTION
			C	Z	P/V	S	N	H	76 543 210		
98	RLD	Fig. 3-8 A, (HL)	-	*	P	*	0	0	11 101 101 01 101 111	2	Contents of low nibble of memory location specified by HL are loaded into high nibble of the same location. High nibble of this memory location is loaded into the low nibble of the accumulator. Low nibble of the accumulator is loaded into the low nibble of the memory byte.
99	RRD	Fig. 3-9 A, (HL)	-	*	P	*	0	0	11 101 101 01 100 111	2	Low nibble of the accumulator is loaded into the high nibble of the memory location specified by HL. High nibble of this memory location is shifted into the low nibble of the same location and the low nibble of this location is shifted into the low nibble of the accumulator.

Table 3-2 Bit Set, Reset and Test Instructions Group.

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS						OP CODE	NO. OF BYTES	OPERATION DESCRIPTION
			C	Z	P/V	S	N	H	76 543 210		
100	BIT b, r	$Z \leftarrow \overline{r_b}$	-	*	?	?	0	1	11 001 011 01 b r	2	The complement of bit b of register r is loaded into the Z flag.
101	BIT b, (HL)	$Z \leftarrow \overline{(HL)_b}$	-	*	?	?	0	1	11 001 011 01 b 110	2	The complement of bit b of memory location specified by HL is loaded into the Z flag.
102	BIT b, (IX + d)	$Z \leftarrow \overline{(IX + d)_b}$	-	*	?	?	0	1	11 011 101 11 001 011 ← d → 01 b 110	4	The complement of bit b of memory location specified by (IX + d) is loaded into flag Z.
103	BIT b, (IY + d)	$Z \leftarrow \overline{(IY + d)_b}$	-	*	?	?	0	1	11 111 101 11 001 011 ← d → 01 b 110	4	The complement of bit b of the memory location specified by (IY + d) is loaded into Z flag.
104	SET b, r	$r_b \leftarrow 1$	-	-	-	-	-	-	11 001 011 11 b r	2	Bit b of operand register r is set.
105	SET b, (HL)	$(HL)_b \leftarrow 1$	-	-	-	-	-	-	11 001 011 11 b 110	2	Bit b of memory location specified by HL is set.
106	SET b, (IX + d)	$(IX + d)_b \leftarrow 1$	-	-	-	-	-	-	11 011 101 11 001 011 ← d → 11 b 110	4	Bit b of memory location specified by (IX + d) is set.

Note: The codes for bit b are given in Table 2-10.

Table 3-2 (Continued) Bit Set, Reset and Test Instructions Group

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS						OP CODE	NO. OF BYTES	OPERATION DESCRIPTION
			C	Z	P/V	S	N	H	76 543 210		
107	SET b, (IY + d)	$(IY + d)_b \leftarrow 1$	-	-	-	-	-	-	11 111 101 11 001 011 ◄— d —► 11 b 110	4	Bit b of memory location specified by (IY + d) is set.
108	RES b, m	$s_b \leftarrow 0$	-	-	-	-	-	-	10	-	Bit b of operand m (as defined by SET b, m above, instructions 104 to 107) is reset. (Note: The code in bits 7 and 6 replaces the corresponding bits 7 and 6 in the last byte of instructions 104 to 107).

3-5 THE JUMP INSTRUCTIONS GROUP

Conditional JUMPS and conditional BRANCHES are performed by testing or sampling the condition of the various flags in the Status Register. The conditions of the various flags are included in the instruction itself and in the Z-80 they are referred to as the cc codes. Table 3-3 below gives these codes and the conditions as well as the flags in the Status Register that are effected.

In the tables that follow Table 3-3, the byte denoted by e-2 is the displacement that is added to the program counter. In the mnemonics, it is labelled as just e. Here e ranges from -126 to +129.

Table 3-3 Conditional JUMP/BRANCH conditions and codes.

cc Code	Status Flag	Condition
000	Z	NZ Non zero
001	Z	Z Zero
010	C	NC No Carry
011	C	C Carry
100	P/V	PO Parity Odd
101	P/V	PE Parity Even
110	S	P Sign Positive
111	S	M Sign Negative

Table 3-4 JUMP Instructions Group.

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS						OP CODE	NO. OF BYTES	OPERATION DESCRIPTION
			C	Z	P/V	S	N	H	76 543 210		
109	JP nn	PC ← nn	-	-	-	-	-	-	11 000 011 ← n → ← n →	3	Operand nn, given in bytes 2 and 3 of the instruction, is loaded into the PC. This is an unconditional JUMP.
110	JP cc, nn	If cc true, then PC ← nn	-	-	-	-	-	-	11 cc 010 ← n → ← n →	3	If condition cc is true, operand nn is loaded into PC. If not, PC is incremented.
111	JR e	PC ← PC + e	-	-	-	-	-	-	00 011 000 ← e-2 →	2	Displacement e-2 is added to PC. This is an unconditional JUMP.
112	JR C, e	If C = 0 then continue, If C = 1, then PC ← PC + e	-	-	-	-	-	-	00 111 000 ← e-2 →	2	If C flag is 0, then the program continues. If not, then the displacement e-2 is added to the PC. This is conditional JUMP.
113	JR NC, e	If C = 1, then continue. If C = 0, then PC ← PC + e	-	-	-	-	-	-	00 110 000 ← e-2 →	2	If C flag is 1, then the program continues. If not, then the displacement e-2 is added to the PC. This is a conditional JUMP.
114	JR Z, e	If Z = 0, then continue. If Z = 1, then PC ← PC + e	-	-	-	-	-	-	00 101 000 ← e-2 →	2	If Z flag is 0, then the program continues. If not, then the displacement e-2 is added to the PC. This is a conditional JUMP.
115	JR NZ, e	If Z=1, then continue. If Z = 0, then PC ← PC + e	-	-	-	-	-	-	00 100 000 ← e-2 →	2	If Z flag is 1, then the program continues. If not, then the displacement e-2 is added to the PC. This is a conditional JUMP.

Table 3-4 (Continued) JUMP Instructions Group.

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS						OP CODE	NO. OF BYTES	OPERATION DESCRIPTION
			C	Z	P/V	S	N	H	76 543 210		
116	JP (HL)	PC ← HL	-	-	-	-	-	-	11 101 001	1	Contents of register pair HL are loaded into the PC. This is an unconditional JUMP.
117	JP (IX)	PC ← IX	-	-	-	-	-	-	11 011 101 11 101 001	2	Contents of index register IX are loaded into the PC. This is an unconditional JUMP.
118	JP (IY)	PC ← IY	-	-	-	-	-	-	11 111 101 11 101 001	2	Contents of index register IY are loaded into the PC. This is an unconditional JUMP.
119	DJNZ e	B ← B - 1 If B = 0, then continue. If B ≠ 0, then PC ← PC + e	-	-	-	-	-	-	00 010 000 ← e-2 →	2	Register B is first decremented. If B is now 0, then the program continues. If B is not 0, then displacement e-2 is added to the PC. This is a conditional JUMP.

3-6 CALL AND RETURN GROUP

The CALL and RETURN instructions are given in Table 3-6. Table 3-5 below gives the starting addresses for special subroutines which are stored in Page 0 of the memory. The 3-bit t code is inserted in the RES p instruction (instruction 126) of Table 3-6. In Table 3-5, the H in the P column indicates that the address is given in the hexadecimal code.

Table 3-5 Starting addresses for special subroutines.

t Code	p (Address in Page 0)
000	00H
001	08H
010	10H
011	18H
100	20H
101	28H
110	30H
111	38H

Table 3-6 CALL and RETURN Instructions Group.

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS						OP CODE	NO. OF BYTES	OPERATION DESCRIPTION
			C	Z	P/V	S	N	H	76 543 210		
120	CALL nn	$(SP - 1) \leftarrow PC_H$ $(SP - 2) \leftarrow PC_L$ $PC \leftarrow nn$	-	-	- -	-	-	-	11 001 101 ← n → ← n →	3	Contents of PC are pushed into the Stack, first high and then the low byte. Operands nn are then loaded into the PC. This is an unconditional BRANCH.
121	CALL cc, nn	If cc true, then $(SP - 1) \leftarrow PC_H$ $(SP - 2) \leftarrow PC_L$ $PC \leftarrow nn$ If cc is false, continue program	-	-	-	-	-	-	11 cc 100 ← n → ← n →	3	If cc condition is true, contents of PC are pushed into Stack and operands nn are loaded into the Stack. If cc is false, PC is incremented and the program continues. This is a conditional BRANCH.
122	RET	$PC_L \leftarrow (SP)$ $PC_H \leftarrow (SP + 1)$	-	-	-	-	-	-	11 001 001	1	Control is returned to the main program upon completion of a subroutine. Contents of (SP) are loaded into low byte of PC. SP is then incremented and contents of (SP + 1) are loaded into high byte of PC.
123	RET cc	If cc is false continue program. If cc is true, $PC_L \leftarrow (SP)$ $PC_H \leftarrow (SP + 1)$	-	-	-	-	-	-	11 cc 000	1	If cc condition is false, subroutine continues. If cc is true, contents of the Stack are popped into the PC and main program is resumed. This is a conditional RETURN.

Table 3-6 (Continued) CALL and RETURN Instructions Group.

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS						OP CODE	NO. OF BYTES	OPERATION DESCRIPTION
			C	Z	P/V	S	N	H	76 543 210		
124	RETI	Return from Interrupt	-	-	-	-	-	-	11 101 101 01 001 101	2	Instruction used at the end of a maskable interrupt service subroutine. Contents of Stack are popped into PC (same as RET instruction). CPU signals the interrupting I/O device that the service subroutine is completed.
125	RETN	Return from non-maskable Interrupt	-	-	-	-	-	-	11 101 101 01 000 101	2	Instruction used at the end of a non-maskable interrupt subroutine Contents of Stack are popped into PC (same as RET instruction). State of IFF2 is loaded into IFF1 so that the maskable interrupts are fully enabled after this instruction.
126	RST p	$(SP - 1) \leftarrow PC_H$ $(SP - 2) \leftarrow PC_L$ $PC_H \leftarrow 0$ $PC_L \leftarrow p$	-	-	-	-	-	-	11 t 111	1	Contents of PC are pushed into the Stack. All zeros are loaded into high byte of PC and address byte p (from Table 3-5) is loaded into low byte of PC. Now PC addresses Page 0 of memory. This restart instruction allows control of program to be given to one of 8 subroutine starting addresses as shown in Table 3-5.

3-7 REVIEW QUESTIONS

3-1 Refer to Figure 3-2 and indicate true or false.

a. ___ This is a left logical shift.

b. ___ This is a left rotate with carry.

c. ___ This is a right rotate with carry.

d. ___ The MSB is shifted into the Carry F/F.

3-2 Pick out the statements that correctly apply to Fig. 3-5.

a. The MSB is shifted into the CY E/F.
b. Right shift arithmetic is performed.
c. The MSB is replicated.
d. After every shift, the bit in the CY F/F is lost.

3-3 Indicate true or false for Fig. 3-8.

a. ___ The high-order nibble of the accumulator is unaltered.

b. ___ The low-order nibble of the accumulator is loaded into the low-order nibble of the HL pair.

c. ___ The low-order nibble of the accumulator is loaded into the high-order nibble of the accumulator.

d. ___ If this instruction is repeated 3 times, the contents of both the accumulator and the HL pair remain unchanged.

3-4 Refer to Fig. 3-3 and fill in the blanks.

a. The _____ of the register is loaded into the _____ position. (MSB; LSB; CY)

b. The _____ of the register is loaded into the CY F/F. (MSB; LSB)

3-5 Figure 3-6 represents a _____ operation. (SLA m; SRA m; SRL m)

3-6 Pick out the correct mnemonic for the instruction where the contents of the accumulator are rotated right one bit and bit 0 is transferred into the CY/F/F.

a. RRCA
b. RLCA
c. RRA
d. RLA

3-7 Contents of the memory location specified by the sum of the index register IX and the displacement d are rotated left one bit and bit 7 is transferred into the CY F/F.

a. The mnemonic for this instruction is _____.

b. This instruction requires _____ bytes.

3-8 Fill in the blanks for the RRD instruction.

a. The _____ nibble of accumulator is loaded into the _____ nibble of memory location specified by HL. (Low; high)

b. The high _____ of this memory location is shifted into the _____ nibble of the same location. Low; high; byte; nibble)

c. The _____ nibble of this memory location is shifted into the _____ nibble of the accumulator. (Low; high)

3-9 Explain in your own words what the symbolic operation

$Z \longleftarrow \overline{r}_b$ means.

_____

_____

_____

3-10 Indicate true or false for the symbolic operation.

$$Z \longleftarrow \overline{(IY + d)_b}$$

a. ___ The complement of register b is transferred into the Z flag.

b. ___ IY + d represents a memory address.

c. ___ The displacement d is provided in the instruction.

d. ___ This is a 2-byte instruction.

e. ___ Bit b of the memory location is inverted and loaded into the Z flag.

3-11 Write down the mnemonic and the symbolic operation for the instruction.

Bit b of operand register r is set.

a. The mnemonic is _____

b. The symbolic operation is _____

3-12 What is the symbolic operation for RES b, m ?

_____

3-13 Fill in the blanks for the mnemonic JP cc, nn.

If condition cc is _____, operand nn is loaded into the _____.

If not, PC is _____. (True; false; PC; HL; IX; IY; incremented; decremented)

3-14 The instruction JR NC, e is a _____ JUMP. (Conditional; unconditional)

3-15 The instruction JP nn is _____JUMP. (Conditional; unconditional)

3-16 In the instruction JR NC, e if the flag is _____, then the program continues. (1; 0)

3-17 In the instruction JP (IX) what happens to the contents of the index register IX?

a. It is loaded into the low byte of the PC.
b. Nothing.
c. It is added to the displacement d.
d. It is loaded into the PC (High and low byte)
e. It is incremented.
f. It is decremented.

3-18 Indicate true or false for the instruction CALL nn.

a. ___ Contents of PC are pushed into the Stack.
b. ___ Stack pointer is decremented twice.
c. ___ Operands nn are loaded into the PC.
d. ___ This is an unconditional BRANCH.

3-19 Indicate true or false for the RET cc instruction.

a. ___ If cc is true, the subroutine continues.

b. ___ If cc is false, contents of Stack are popped into the PC.

c. ___ The main program is then terminated.

d. ___ This is a conditional RETURN.

3-20 The instruction RETI is used at the _____ of a _____ interrupt service subroutine. (Beginning; end; maskable; non-maskable)

4 THE Z-80 INSTRUCTION SET & INTERRUPT RESPONSE

CHAPTER OBJECTIVES

The objectives of this chapter are as follows:

1. To present the INPUT group of instructions.
2. To present the OUTPUT group of instructions.
3. To present the various instructions involved in the operation of the Z-80/CPU.
4. To present the instructions associated with the maskable interrupt feature.
5. To present the instructions associated with the interrupt modes available to the Z-80.
6. To describe the Z-80 interrupt system, including the operation and function of the interrupt enable-disable flip-flop.
7. To present and discuss the CPU responses to both the maskable and the non-maskable interrupts.

TEXTBOOK REFERENCES

For reviewing material in the main hardcover textbook, relevant to the topics covered in this chapter, the following chapters and/or sections are suggested:

1. For INPUT/OUTPUT instructions - Sec. 4-2.6
2. For HALT instruction - Sec. 4-2.3.1
3. For complement and set instructions - Sec. 4-2.5
4. For DAA instruction - Sec. 2-4.5 and 2-4.6

5. For interrupt system operation - Chapter 9

4-1 INTRODUCTION

This chapter is a continuation of Chapters 2 and 3 and it completes the Z-80 Instruction set. In this chapter we will cover the following groups of instructions:

1. The Input/Output group.

2. The basic CPU instructions group.

In this chapter we also give a fairly extensive discussion of the interrupt processes as used in the Z-80 system.

4-2 THE I/O INSTRUCTION GROUP

Table 4-1 The Input Instructions Group.

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS						OP CODE	NO. OF BYTES	OPERATION DESCRIPTION
			C	Z	P/V	S	N	H	76 543 210		
127	IN A, (n)	A ← (n)	-	-	-	-	-	-	11 011 011 ← n →	2	1. Operand n → (A0 - A7) lines. (n is now the address of one of the 256 possible I/O ports). 2. Also, the contents of the accumulator appear on upper half of the address bus ACC → (A8 - A15) 3. As a result, one byte from the I/O is placed on the Data Bus. I/O → (D0-D7) → ACC
128	IN r, (C)	r ← (C)	-	*	P	*	0	*	11 101 101 01 r 000	2	1. Contents of register C → (A0 - A7) lines. 2. Contents of register B → (A8 - A15) lines. 3. I/O → (D0-D7) → ACC Note: If r = 110, only the flag bits are affected.
129	INI	(HL) ← (C) B ← B - 1 HL ← HL + 1	?	* @	?	?	1	?	11 101 101 10 100 010	2	1. Contents of register C → (A0 - A7) lines. 2. Register B is used as a byte Counter. B → (A8 - A15). 3. I/O → (D0 - D7) → CPU 4. (HL) → (A0 - A15) lines. 5. Finally, B is decremented and HL is incremented.

Table 4-1 (Continued) The Input Instructions Group.

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS						OP CODE	NO. OF BYTES	OPERATION DESCRIPTION
			C	Z	P/V	S	N	H	76 543 210		
130	INIR	(HL) ← (C) B ← B - 1 (HL) ← HL + 1 Repeat until B = 0	?	*	?	?	1	?	11 101 101 10 110 010	2	1. Contents of register C → (A0 - A7) lines. 2. Register B is byte counter and B → (A8 - A 15) lines. 3. I/O → (D0 - D7) → CPU 4. (HL) → (A0 - A15) lines. 5. HL is incremented and B is decremented. 6. If B = 0, instruction is terminated. 7. If B ≠ 0, PC - 2 → PC and instruction is repeated.
131	IND	(HL) ← (C) B ← B - 1 HL ← HL - 1	?	* @	?	?	1	?	11 101 101 10 101 010	2	1. C → (A0 - A7) lines. 2. Register B is byte counter and B → (A8 - A15) lines. 3. I/O → (D0 - D7) → CPU 4. (HL) → (A0 - A15) lines. 5. B and HL are decremented.
132	INDR	(HL) ← (C) B ← B - 1 HL ← HL - 1 Repeat until B = 0	?	*	?	?	1	?	11 101 101 10 111 010	2	1. C → (A0 - A7) lines. 2. Register B is byte counter and B → (A8 - A15) lines. 3. I/O → (D0 - D7) → CPU 4. (HL) → (A0 - A15) lines. 5. B and HL are decremented. 6. If B = 0, instruction is terminated. 7. If B ≠ 0, PC - 2 → PC and instruction is repeated.

Table 4-2 The Output Instructions Group.

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS						OP CODE	NO. OF BYTES	OPERATION DESCRIPTION
			C	Z	P/V	S	N	H	76 543 210		
133	OUT (n), A	(n) ← A	-	-	-	-	-	-	11 010 011 ← n →	2	1. n → (A0 - A7) lines 2. (ACC) → (A8 - A15) lines 3. (ACC) → (D0 - D7) → I/O device
134	OUT (C), r	(C) ← r	-	-	-	-	-	-	11 101 101 ← r →	2	1. (C) → (A0 - A7) lines 2. (B) → (A8 - A15) lines 3. (r) → (D0 D7) → I/O device
135	OUTI	(C) ← (HL) B ← B - 1 HL ← HL + 1	?	*	?	?	1	?	11 101 101 10 100 011	2	1. (HL) → (A0 - A15) lines 2. HL selects a byte in addressed memory location. 3. Memory byte → CPU (temporarily stored) 4. B - 1 → B 5. (C) → (A0 - A7) lines 6. (B - 1) → (A8 - A15) lines 7. CPU (output byte) → (D0 - D7) → I/O device 8. HL + 1 → HL

@

Table 4-2 (Continued) The Output Instructions Group.

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS						OP CODE	NO. OF BYTES	OPERATION DESCRIPTION
			C	Z	P/V	S	N	H	76 543 210		
136	OUTIR	(C) ← (HL) B ← B - 1 HL ← HL + 1 Repeat until B = 0	?	1	?	?	1	?	11 101 101 10 110 011	2	1. (HL) → (A0 - A15) lines 2. HL selects a byte in memory. 3. Memory byte → CPU. (temporarily stored) 4. B - 1 → B 5. (C) → (A0 - A7) lines 6. (B - 1) → (A8 - A15) lines 7. CPU (output byte → (D0 - D7) → I/O device 8. HL + 1 → HL 9. IF B ≠ 0, PC -2 → PC and instruction is repeated. 10. If B = 0, instruction is terminated.
137	OUTD	(C) ← (HL) B ← B - 1 HL ← HL - 1	?	* @	?	?	1	?	11 101 101 10 101 011		1. (HL) → (A0 - A15) lines 2. HL selects a byte in memory. 3. Memory byte → CPU (temporarily stored) 4. B - 1 → B 5. (C) → (A0 - A7) lines 6. (B - 1) → (A8 - A15) lines 7. CPU (output byte) → (D0 - D7) → I/O device. 8. HL - 1 → HL

Table 4-2 (Continued) The Output Instructions Group.

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS						OP CODE	NO. OF BYTES	OPERATION DESCRIPTION
			C	Z	P/V	S	N	H	76 543 210		
138	OUTDR	(C) ← (HL) B ← B - 1 HL ← HL - 1 Repeat until B = 0	?	*	?	?	1	?	11 101 101 10 111 011	2	1. HL ⟶ (A0 - A15) lines 2. HL selects a byte in memory. 3. Memory byte ⟶ CPU (temporarily stored) 4. B - 1 ⟶ B 5. (C) ⟶ (A0 - A7) lines 6. (B - 1) ⟶ (A8 - A15) lines 7. CPU (Output byte) ⟶ (D0 - D7) ⟶ I/O device. 8. HL - 1 ⟶ HL 9. If B =/ 0, PC - 2 ⟶ PC and instruction is repeated. 10. If B = 0, instruction is terminated.

4-3 THE BASIC CPU INSTRUCTION GROUP

Table 4-3 Basic CPU Instructions Group.

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS						OP CODE	NO. OF BYTES	OPERATION DESCRIPTION
			C	Z	P/V	S	N	H	76 543 210		
139	DAA	ACC contents are converted into packed BCD for BCD add and subtract	*	*	P	*	-	*	00 100 111	1	Conditionally adjusts ACC for BCD addition and subtraction. Additions involve ADD, ADC and INC instructions. Subtractions involve SUB, SBC, DEC and NEG instructions.
140	CPL	$A \leftarrow \overline{A}$	-	-	-	-	1	1	00 101 111	1	Contents of ACC are complemented.
141	NEG	$A \leftarrow 0 - A$	*	*	V	*	1	1	11 101 101 01 000 100	2	ACC contents are negated (2s complement) by subtracting (ACC) from 0.
142	CCF	$CY \leftarrow \overline{CY}$	*	-	-	-	0	?	00 111 111	1	The CY F/F in register F is complemented.
143	SCF	$CY \leftarrow 1$	1	-	-	-	0	0	00 110 111	1	The CY F/F in register F is set to 1.
144	NOP	No operation	-	-	-	-	-	-	00 000 000	1	No operation is performed by CPU during this machine cycle.
145	HALT	CPU is halted	-	-	-	-	-	-	01 110 110	1	CPU operation is suspended until an interrupt or reset comes along. CPU executes NOP instructions which keeps on refreshing dynamic memory chips.
146	DI	$IFF \leftarrow 0$	-	-	-	-	-	-	11 110 011	1	Maskable interrupt feature is disabled by resetting IFF1 and IFF2 flip-flops.

Table 4-3 (Continued) Basic CPU Instructions Group.

NO.	MNEMONICS	SYMBOLIC OPERATIONS	STATUS FLAGS						OP CODE	NO. OF BYTES	OPERATION DESCRIPTION
			C	Z	P/V	S	N	H	76 543 210		
147	EI	IFF ← 1	-	-	-	-	-	-	11 111 011	1	Maskable interrupt feature is enabled by setting IFF1 and IFF2 flip-flops to the 1 side.
148	IM 0	Set interrupt Mode 0	-	-	-	-	-	-	11 101 101 01 000 110	2	Mode 0 interrupt is activated. The interrupting device sends the instruction to be executed to the CPU via the data lines (D0 - D7), one byte at a time.
149	IM 1	Set interrupt Mode 1	-	-	-	-	-	-	11 101 101 01 010 110	2	Mode 1 interrupt is activated. CPU responds to interrupt by executing a restart to location 0038H in memory.
150	IM 2	Set interrupt Mode 2	-	-	-	-	-	-	11 101 101 01 011 110	2	Mode 2 interrupt is activated. Interrupting I/O supplies 8-bit vector for low-byte of indirect address pointer. Register I supplies the high-order byte of the pointer. This mode provides an indirect CALL to any memory location for the starting address of the service subroutine.

NOTE - Table 4-3 completes the Z-80 instruction set with instruction IM 2 which brings it to a total of 150 instructions. In Section 2-1 it was mentioned that the Z-80 has a total of 157 instructions. Students may wonder what happened to the remaining 8 instructions.

This discrepancy is readily explained by looking at the RST p instruction (instruction 126 in Table 3-6). Notice that the OP code of this instruction contains the 3-bit t code which defines the starting addresses for special subroutines (See Table 3-5.) Since 8 such starting addresses are possible, instruction 126 in Table 3-6 is the general format for 8 such instructions which brings the total available instructions in the Z-80 set to 157.

4-4 THE INTERRUPT SYSTEM

4-4.1 The Objectives

1. Interrupts are initiated by external I/O peripherals or other devices to demand servicing of the particular service subroutine.

2. The CPU current instruction is completed and the CPU operation is suspended in an orderly fashion.

3. Service subroutines normally involve transfer of data between peripherals and CPU or status and control information between the two.

4. Upon completion of the service subroutine, CPU resumes execution of the main program.

4-4.2 The Non-Maskable Interrupt (NMI)

1. This type of interrupt is designed into the hardware of the CPU logic and is not under control of the programmer.

2. The programmer cannot selectively disable or enable this interrupt.

3. The NMI is accepted, and serviced, by the CPU whenever it arrives. It has the highest priority and can override all other operations.

4. The NMI is reserved for very critical functions, such as an impending power failure, whose demand for service cannot be delayed.

4-4.3 The Maskable Interrupt (INT)

1. The INT can be selectively deactivated by the programmer.

2. When the INT is deactivated, any incoming interrupts will be ignored by the CPU.

3. This feature is generally used by the programmer when certain critical instructions in the program are being executed. Usually such instructions involve critical timing restraints which cannot be interrupted.

4. When execution of these critical instructions is completed, the INT feature is reactivated.

4-4.4 The INT Enable-Disable Flip-Flop

1. The Z-80/CPU has a flip-flop, IFF1, which is used to control the acceptance or rejection of the maskable interrupt (INT).

2. The DI instruction (See Table 4-3) resets IFF1 to the 0 side, and this forces the CPU to ignore all incoming INT requests.

3. The EI instruction (see Table 4-3) sets IFF1 to the 1 side. Now the CPU will recognize and honor INT requests from the I/O devices.

4. Notice one peculiarity of the EI instruction. When it is being executed, if any INT comes along it will only be accepted after completion of the instruction following the EI instruction. This one instruction delay is particularly useful if the instruction after the EI instruction is a return from subroutine (RET).

5. In the Z-80/CPU there are really two enable-disable flip-flops. The second one is labelled IFF2. Both these flip-flops work in unison for the maskable interrupt situations.

6. The function of IFF2 is to save the status of IFF1 when a non-maskable interrupt (NMI) happens. Here is how it works:

   a. When an NMI is recognized, IFF1 is immediately reset.
   b. This locks out any other INT from coming in the CPU while the NMI is being serviced.
   c. During the NMI servicing, IFF2 is not disturbed and retains the prior status of IFF1 (either a 0 or a 1).

7. The last instruction in the NMI service subroutine is a return from non-maskable interrupt (RETN) instruction (See Table 3-6).

8. The RETN instruction copies the status of IFF2 into IFF1 thereby returning IFF1 to its pre-NMI status automatically.

4-4.5 CPU Response to Non-Maskable Interrupts (NMI)

1. The CPU accepts the NMI at all times.

2. The CPU will ignore the next instruction it has fetched and instead goes to memory location 0066H and performs an operation similar to a restart operation.

3. Notice that the memory location 0066H is not included in the 8 software-controlled restart addresses located in Page 0 of the memory. (See Table 3-5.)

4-4.6 CPU Response to Maskable Interrupts (INT)

The Z-80 has three different operational modes for responding to NMI requests. The modes can be changed by means of program instructions. (See Table 4-3.) Each of these modes is briefly described below:

4-4.6.1 Mode 0

1. In this mode, the interrupting device sends an 8-bit instruction to the CPU on the data bus lines (D0 - D7).

2. The CPU does not execute the next instruction, fetched from the memory but instead executes the instruction sent on the data bus.

3. This instruction could be a 3-byte CALL (i.e. a branch) instruction to some other location in the memory.

4. In executing this instruction, the CPU automatically inserts two wait states in the special M cycle. This will be explained later in Sec. 5-6 with Fig. 5-9. This allows sufficient time for the daisy chain to respond and the interrupting device to identify itself.

4-4.6.2 Mode 1

1. In this mode the CPU responds to an INT by automatically branching to memory location 0038H which is the starting address of the subroutine.

2. This response is identical to the NMI response, described in Sec. 4-4.5 except that the subroutine starting address is 0038H instead of 0066H.

3. One difference is that, unlike the NMI response, the CPU automatically inserts two wait states for reasons that were previously discussed. See and compare the two timing charts in Fig. 5-9 and Fig. 5-10.

4-4.6.3 Mode 2

1. This is the most powerful and versatile interrupt response. Its primary objective is to give the user the ability to branch to any location within the memory that can be addressed essentially with a 15-bit address for the starting address of the subroutine.

2. An indirect addressing mode is used, and the address is formed as explained below.

3. The programmer constructs a 16-bit address table and pre-stores it in the memory. The 2-byte addresses are the starting addresses of the various service subroutines.

4. The Interrupt Vector Register I, is previously stored with a desired byte which forms the high-order byte of the concatenated 16-bit address.

5. The low-order of the concatenated 16-bit address is supplied by the interrupting device.

6. Since the concatenated address is a 2-byte address, this fact is indicated by bit 0 of the low-order byte which is always a 0.

7. The 16-bit address thus formed is used as a pointer which points to the previously-stored table of starting addresses. This table contains the true or effective starting addresses of the desired service subroutines.

8. This is a branch operation which requires 19 clock periods as shown below:

   7 - Fetch low-order byte from interrupting device.
   6 - Push PC contents into the Stack.
   6 - Fetch the effective starting address from the table.

9. Figure 4-1 is a graphical representation of Mode 2 interrupt operation.

4-5 REVIEW QUESTIONS

4-1 _____ In the Z-80 system, how many I/O ports is it possible to address? (8; 16; 32; 64; 256; 512; 1024)

4-2 Refer to Table 4-1 and indicate true or false for the instruction

IN a, (n)

a. ___ Operand n is the address of one of the ports

b. ___ The CPU places n on the D0 - D7 lines.

c. ___ The accumulator places its contents on the lower half of the address bus.

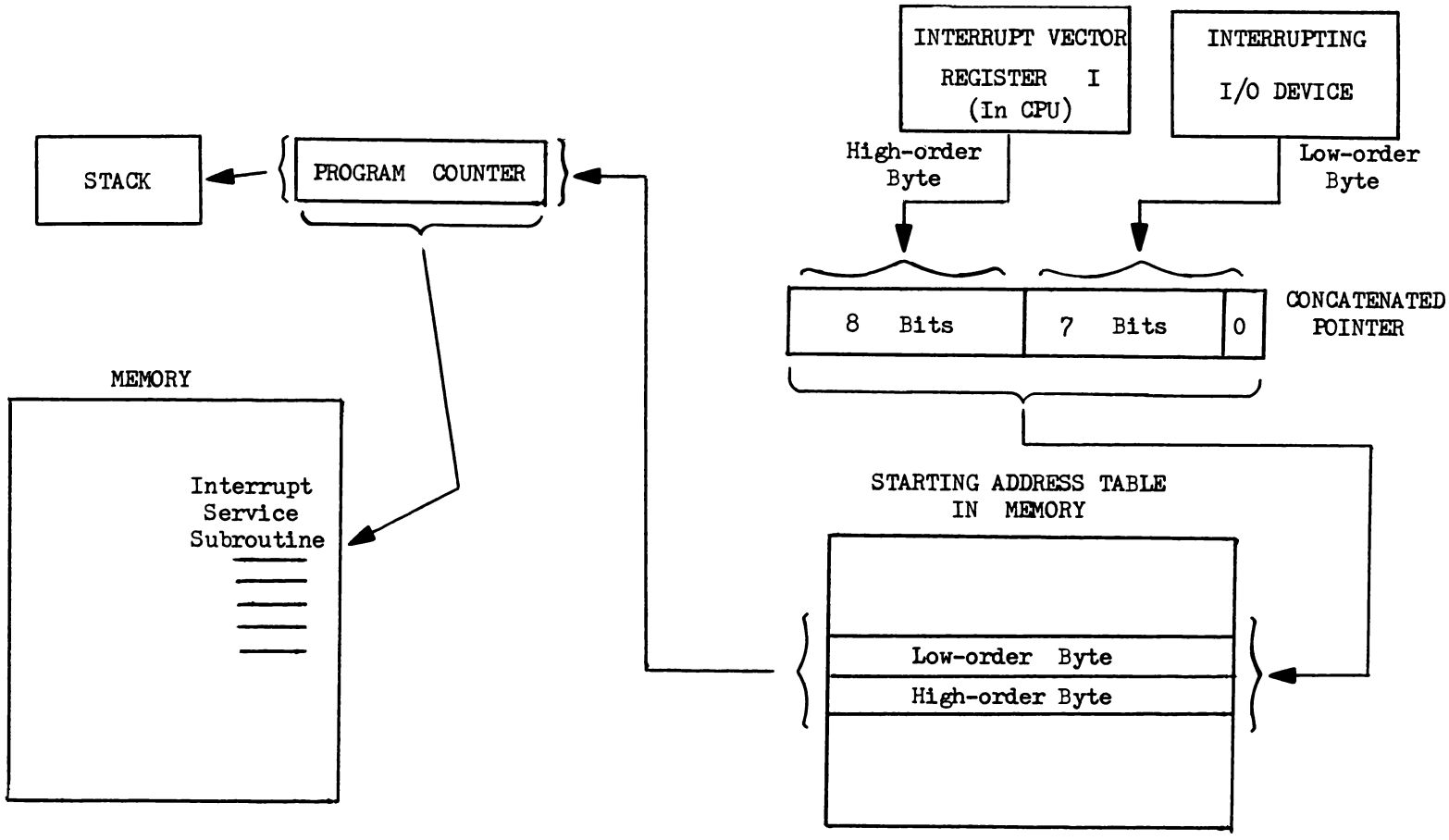


Fig. 4-1 Graphical representation of the Mode 2 Interrupt Response.

d. ___ One byte from the I/O is loaded into the accumulator via the data bus.

4-3 The following statements apply to the IN r, (C) instruction. Fill in the blanks.

a. The contents of register _____ are put on the A8 - A15 lines. (A; B; C; D; E; H; L)

b. The contents of register _____ are put on the A0 - A7 lines. (A; B; C; D; E; H; L)

c. The I/O device loads its 8-bit word into the _____. (Accumulator; B register; C register)

4-4 In the INI instruction _____ is used as a byte counter. (Register A; register B; register C)

4-5 In the INI instruction _____ is decremented and register pair _____ is incremented. (A; B; C; D; I; BC; DE; HL)

4-6 Indicate true or false for the INIR instruction.

a. ___ The instruction is repeated until the contents of BC are 0.

b. ___ The contents of the byte counter are put on bus lines A8 - A15.

c. ___ The contents of the HL pair are put on bus lines A0 - A7.

d. ___ If B $\neq$ 0, the instruction is terminated.

e. ___ If B $\neq$ 0, the contents of PC is decremented once.

4-7 In the IND instruction, the contents of both register B and the register pair HL are _____ . (Incremented; decremented)

4-8 Indicate true or false for the INDR instruction.

a. ___ Register B is the byte counter.

b. ___ B and HL are both incremented.

c. ___ If B = 0, the instruction is terminated.

d. ___ Instruction is repeated until B = 0.

4-9 The instruction OUT (C), r outputs the contents of register _____ to the I/O on the _____ lines. (m; n; r; B; C; A0 - A7; A8 - A15; D0 - D7)

4-10 The following statements relate to the OUTI instruction. Indicate true or false.

a. ___ HL selects a byte in the addressed memory location.

b. ___ The memory byte is temporarily stored in the CPU.

c. ___ Register B is incremented.

d. ___ Contents of register B are put on the A8 - A15 lines.

e. ___ Contents of register C are put on the A0 - A7 lines.

4-11 Fill in the blanks for the OUTIR instruction.

a. If _____, the instruction is terminated. (B $\neq$ 0; B = 0)

b. If _____, the instruction is repeated until _____ . (B = 0; B $\neq$ 0; PC = 0; PC $\neq$ 0).

c. The program counter is _____ twice. (incremented; decremented)

4-12 In the OUTD instruction, the word to be outputted to the I/O device is originally located in the _____. (accumulator; B register; memory; HL register)

4-13 The DAA instruction applies to the _____ arithmetic operations. (BCD; hexadecimal; binary)

4-14 Which of the following mnemonics is used to complement the contents of the accumulator?

a. CPL
b. CCF
c. SCF
d. DI
e. None of the above

4-15 The SCF instruction _____ the carry F/F. (Sets; resets; complements)

4-16 Indicate true or false for the HALT instruction.

a. ___ Dynamic memory chips are refreshed.

b. ___ NOP instructions are executed.

c. ___ HALT state continues until an interrupt comes in.

4-17 The maskable interrupt feature is disabled by the _____ instruction which

_____ the IFF1 and the IFF2 flip-flops. (EI; DI; resets; sets)

4-18 Identify the interrupt mode in which the interrupting I/O device sends the struction to be executed to the CPU.

a. Mode 0
b. Mode 1
c. Mode 2

4-19 Identify the interrupt mode which provides an indirect branch to a memory location for the starting address of the service subroutine.

a. Mode 0
b. Mode 1
c. Mode 2

4-20 Which type of interrupt mode can be selectively deactivated?

a. NMI
b. INT

5 THE Z-80/CPU TIMING AND CONTROL

CHAPTER OBJECTIVES

The objectives of this chapter are as follows:

1. To explain the fundamental Z-80 instruction cycle and the concept of M cycles.
2. To present and analyze the OP code fetch cycle with and without WAIT states.
3. To present and discuss the memory read and write cycles with and without WAIT states.
4. To present and examine the I/O read and write cycles with and without WAIT states.
5. To examine bus request and acknowledge cycles and their respective timings.
6. To present and discuss the interrupt request and acknowledge cycles for both the maskable and the non-maskable interrupts.
7. To present the timing involved in an exit from HALT cycle.

TEXTBOOK REFERENCES

For reviewing material in the main hardcover textbook, relevant to the topics covered in this chapter, the following chapters and/or sections are suggested:

1. For general discussion on machine cycles - Secs. 3-2.1 and 3-2.2
2. For fixed instruction cycle - Sec. 3-2.3.1
3. For variable instruction cycle - Sec. 3-2.3.2
4. For general review of interrupt I/O - Chapter 9
5. For maskable interrupts - Sec. 9-3

5-1 THE FUNDAMENTAL INSTRUCTION CYCLE

As previously explained in Section 1-3 (paragraphs 3, 4 and 5) and Table 1-5 in Chapter 1, the Z-80 uses a single-phase clocking system for controlling and synchronizing all its operations. The Z-80 instructions execute the following five basic operations:

1. Memory read,
2. Memory write,
3. External device (I/O) read,
4. External devices (I/O) write, and
5. Interrupt acknowledge

In the Z-80/CPU, the clock period is defined as the time interval between the leading edge of one positive-going edge of the clock pulse and the leading edge of the next positive-going edge of the clock pulse. It is referred to as the T cycles as shown in Fig. 5-1. Each of the above-mentioned 5 basic operations normally takes from 3 to 6 clock periods to complete. However, the Z-80/CPU is designed to be flexible in its operations so that the time required to perform these operations could be lengthened beyond the 6 clock periods in order to synchronize the CPU with the slower peripherals.

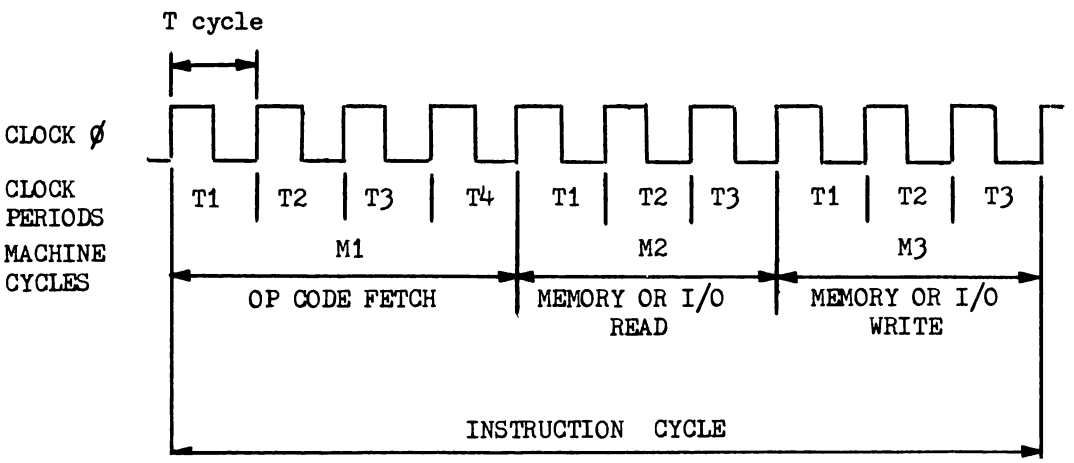


Fig. 5-1 Z-80/CPU basic instruction cycle timing.

As shown in Fig. 5-1, the instruction cycle is broken down into sub-cycles, referred to as the M cycles. Each M cycle consists of several different clock periods or T cycles. Thus, each instruction cycle will consist of a series of M cycles and each M cycle will consist of a different number of T cycles. Figure 5-1 is merely an example of a typical instruction. The M1 subcycle fetches the OP code of the next

instruction to be executed. Normally, M1 consists of 4-6 T cycles. However, M1 could be lengthened by a WAIT signal from an external device (see Table 1-3). M2 and M3 (as shown in Fig. 5-1) are machine cycles that merely transfer data between the CPU and the memory or I/O devices. M2 and M3 are normally 3-5 T cycles long but they could also be lengthened by the WAIT signals to accomodate slower external devices.

Figure 5-1 only shows a fundamental timing diagram. The specific timing for each operation is of course different. In this chapter we will describe and discuss the timing charts for the following 7 specific operations. The diagrams are intended to show operations with and without the WAIT states. It is recommended that students refer to Tables 1-1 through 1-4 where the functions of the various CPU signals are described. The 7 specific operations are:

1. The OP code fetch cycle (M1),

2. Memory read and write cycles,

3. I/O read and write cycles,

4. Bus request/acknowledge cycles.

5. Maskable Interrupt request/acknowledge cycles,

6. Non-maskable Interrupt request/acknowledge cycles, and

7. Exit from a Halt instruction cycle.

5-2 THE OP CODE FETCH CYCLE

5-2.1 Without WAIT States

1. Figure 5-2 shows the timing of the various CPU signals involved in fetching the OP code of the instruction. Notice that in Fig. 5-1 the clock pulses are shown as perfect square waves for simplicity. Of course the leading and trailing edges of each pulse have associated with them corresponding rise and fall times. In the rest of this chapter, the clock pulses are shown with their rise and fall times. Also, all the other signals are referenced to the 50% point of the rise and fall times.

2. Figure 5-2 shows the M1 cycle and the status of each signal during the 4 clock periods, T1 - T4. The first step in fetching the OP code is to transfer the contents of the PC, i.e. the instruction address, to the memory. In Fig. 5-2 this is shown. The address bus A0 - A15 is activated during T1 and T2 clock periods. Note that the two waveforms as shown indicate that some of the address lines will be high and others low, depending upon the combination of 1s and 0s in the address.

3. During the T3 and T4 clock periods, the lower 7 lines of the address bus are

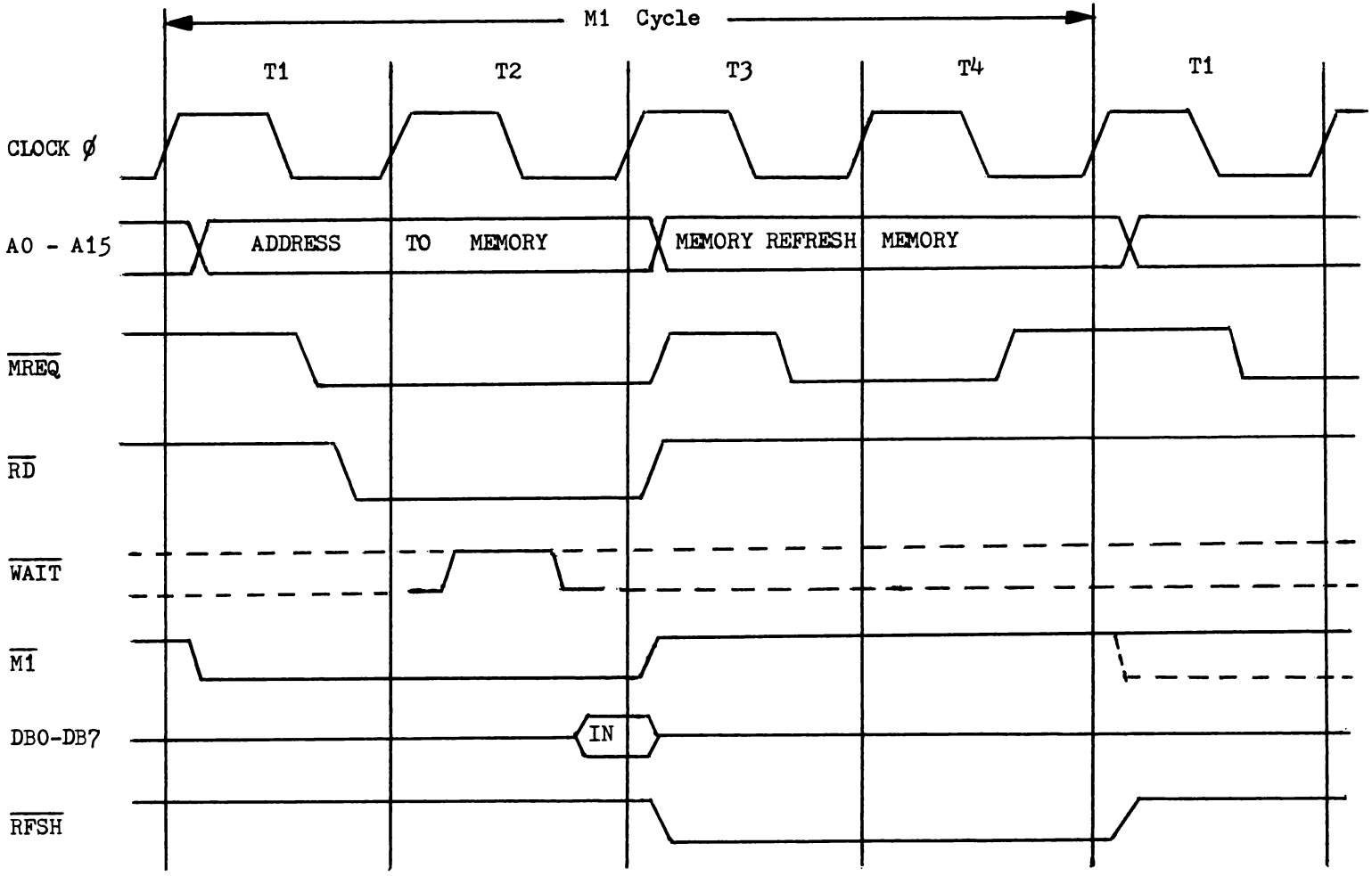


Fig. 5-2 CPU timing for instruction OP code fetch without WAIT.

5-3.2 With WAIT States

1. Figure 5-5 shows two WAIT state cycles, TW1 and TW2, introduced by two $\overline{\text{WAIT}}$ pulses during T2 and TW1 respectively. The operation due to these two pulses is identical to that previously described for the OP code fetch cycle.

2. In Fig. 5-5 we have shown separate data bus situations for the read and write cycles using the same $\overline{\text{WAIT}}$ signals. Of course it is impossible to have a read and a write operation simultaneously. Figure 5-5 is drawn this way for convenience only.

3. Notice that in the read operation, data are placed on the data bus D0 - D7 only during the T3 clock period. For the write operation, the CPU maintains the data bits on the data bus from the second half of T1 and T2, through the wait states, and through T3. This gives the memory ample time to write the data into the accessed locations.

5-4 I/O READ AND WRITE CYCLES

5-4.1 Without WAIT states

1. The I/O read and write operations are similar to the memory read/write operations in most respects but there are also differences as explained below.

2. Figure 5-6 shows the I/O read and write operations cycles. Note that the $\overline{\text{IORQ}}$ is now used instead of the $\overline{\text{MREQ}}$ signal as in the previous case.

3. The first obvious difference is that Fig. 5-6 includes a wait state identified as the TWO cycle. This state is intentionally designed and inserted into the system for the following reasons:

   a. The $\overline{\text{IORQ}}$ signal is activated during the T2 cycle. If the CPU were to use the same $\overline{\text{WAIT}}$ line during T2, then there would be very little time available between these two signals being activated.

   b. Consequently, this very short period of time would not be sufficient for the I/O port to decode its address and activate the $\overline{\text{WAIT}}$ line if a genuine wait were required.

   c. I/O devices using the MOS technology are inherently slower than the CPU. The short time between these two signals would only aggravate the speed incompatibility problem even more.

4. During TWO, the CPU samples the $\overline{\text{WAIT}}$ line as shown in Fig. 5-6.

5. During the read operation, the RD signal is used to gate the contents of the particular port that was addressed on to the data bus lines D0 - D7.

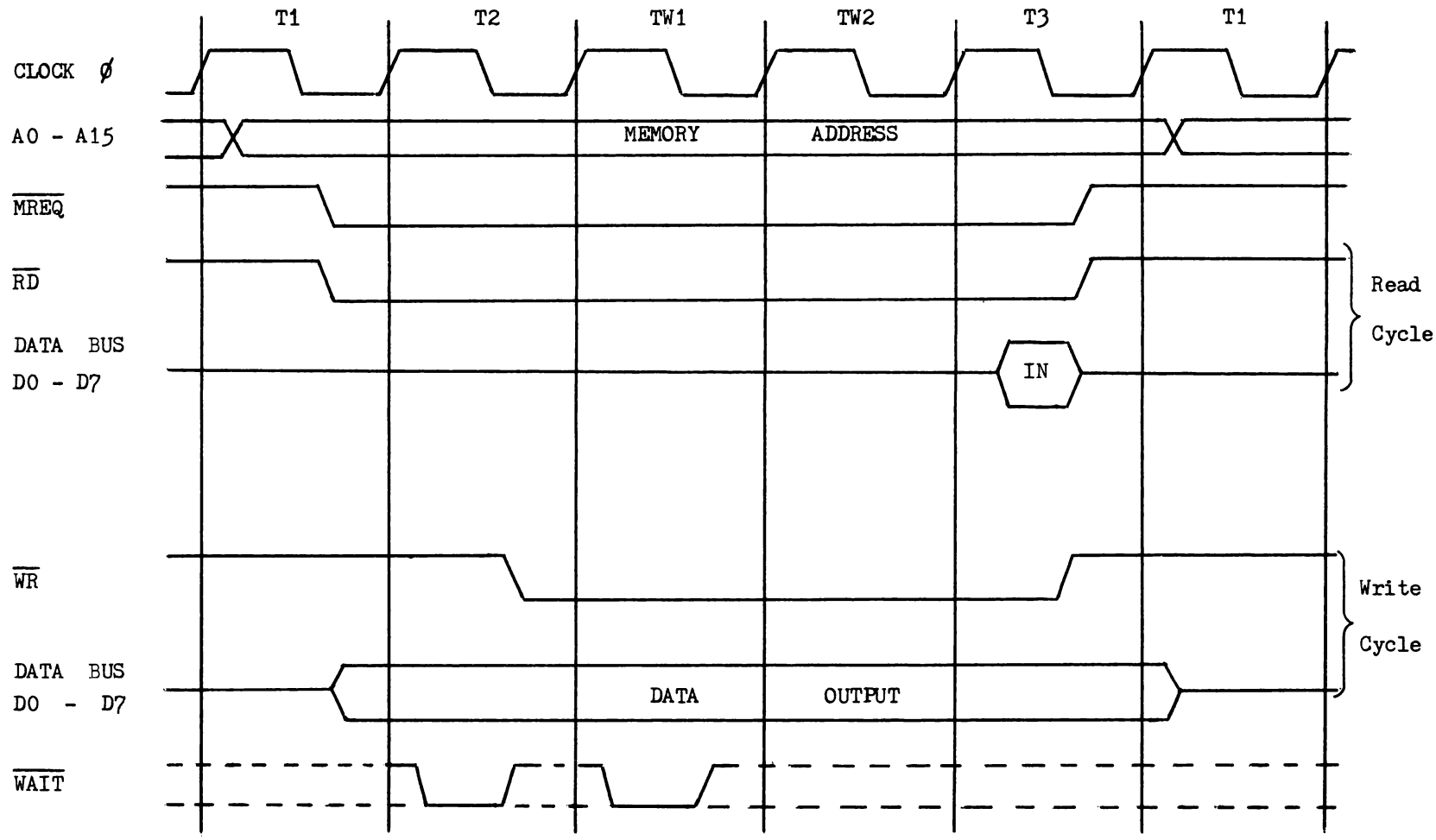


Fig. 5-5 Memory read/write cycle timing with WAIT.

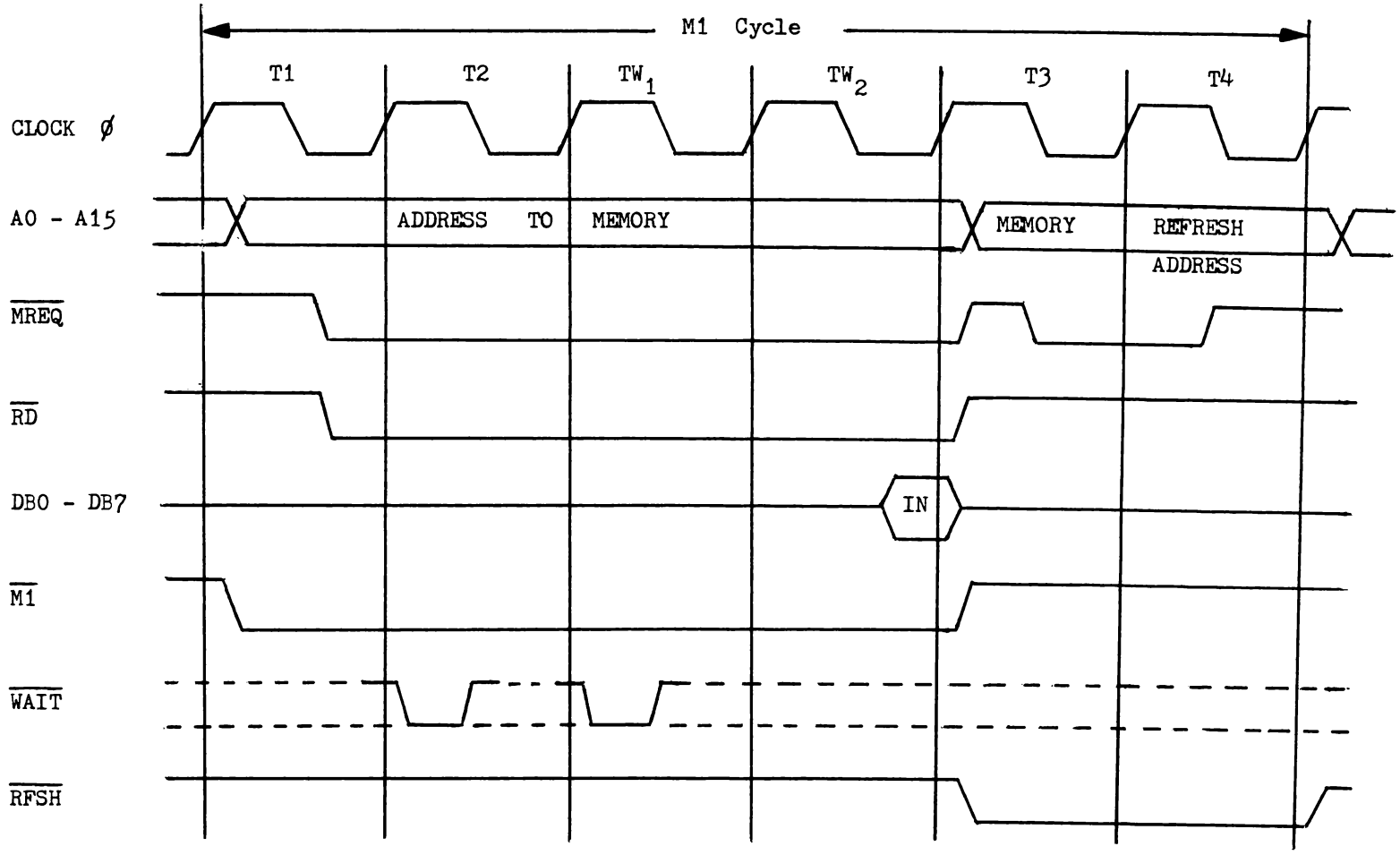


Fig. 5-3 CPU timing for instruction OP code fetch with WAIT.

4. During TW1, the CPU again samples the $\overline{WAIT}$ line. If this signal is active, the CPU goes into another wait cycle, identified as TW2 in Fig. 5-3.

5. Every subsequent wait cycle, TW, the line is checked for its status. When the WAIT is sampled and found to be high, clock periods T3 and T4 are resumed and the normal functions are executed.

6. This technique of status checking allows the Z-80/CPU to work effectively with memory chips requiring much longer access times.

5-3 MEMORY READ AND WRITE CYCLES

5-3.1 Without WAIT States

1. Because an OP code fetch operation is a memory access operation, the memory read/write cycles are very similar to the previously-described fetch operation for the OP code (Section 5-2), and so will not be repeated here.

2. Both the memory read and write cycles each require 3 clock periods. They are shown in Fig. 5-4 as two separate M cycles.

3. The functions and operations of both the $\overline{MREQ}$ and $\overline{RD}$ signals are identical to those for the OP code fetch operation.

4. Of course, the address of the memory location to be accessed is first placed on the address bus A0 -A15.

5. During both the read and the write operations, the $\overline{MREQ}$ line is active during the second half of T1, all of T2 and the first half of T3. This allows the address bus to stabilize. This is especially important during the write cycle. This then allows the $\overline{MREQ}$ to be directly used as a chip enable signal in dynamic memory chip.

6. During the memory write cycle, note that the $\overline{WR}$ signal is active during the second half of T2 and the first half of T3. The $\overline{WR}$ signal is issued by the CPU to inform the memory that the CPU has placed valid data on the Data Bus and that it should be accepted by the memory.

7. Because of the time span during which $\overline{WR}$ is active (see Fig. 5-4), it is possible to use it as R/$\overline{W}$ signal and so can be used for almost any semiconductor memory chips.

8. Also notice that $\overline{WR}$ goes high one-half clock period (i.e. during the second half of T3) before the contents of either the address bus or the data bus are changed. This makes the Z-80 flexible enough to meet the overlap limitations and makes it suitable for use with many different types of memory chips.

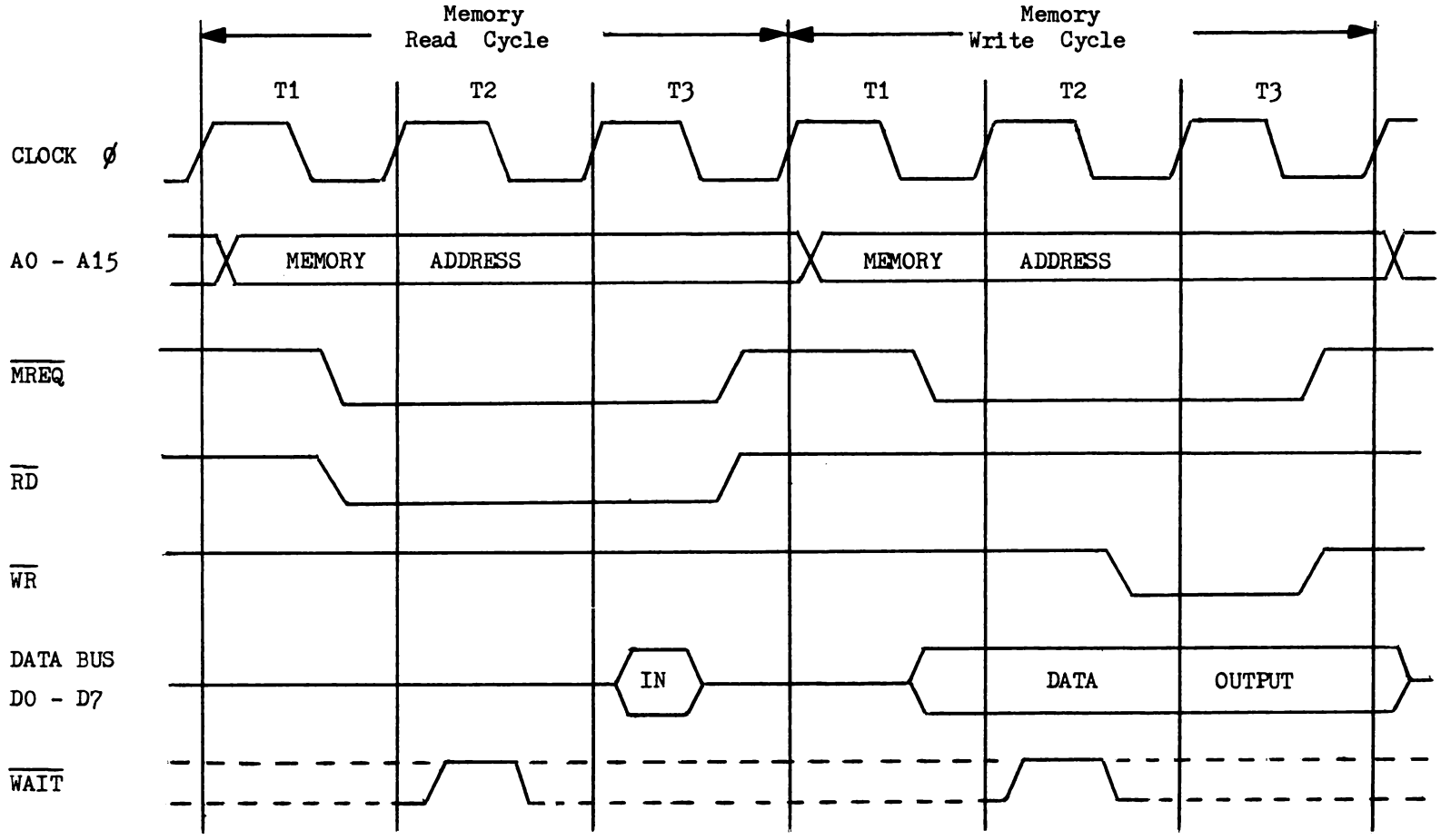


Fig. 5-4 Memory read/write cycle timing without WAIT.

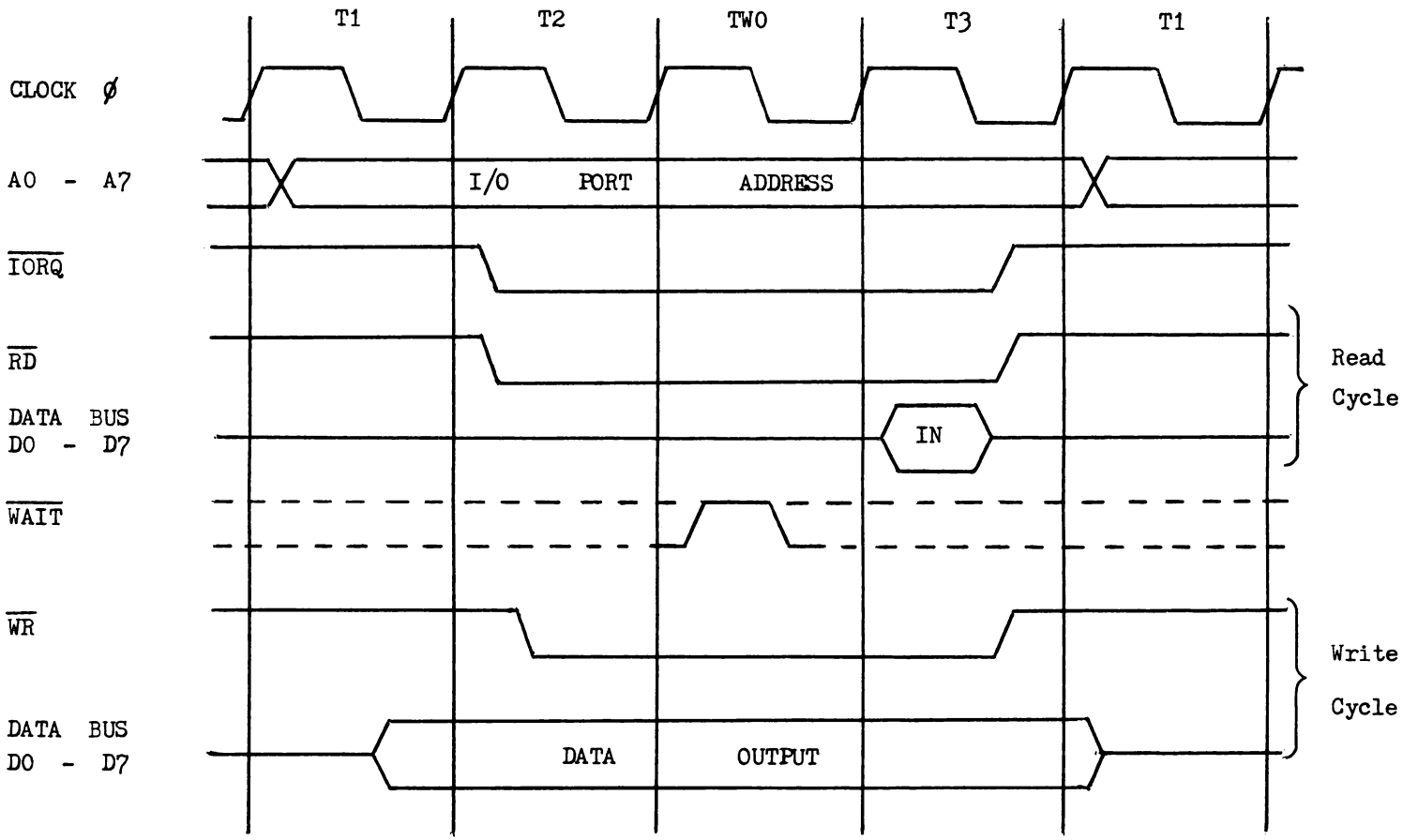


Fig. 5-6 I/O read and write cycle timing without WAIT states.

6. During the write operation, the $\overline{WR}$ signal is used as a clock signal to the addressed I/O port. It is used to clock the available bits on the data bus lines to be inputted into the I/O device.

5-4.2 With WAIT States

1. Figure 5-7 shows the I/O read/write operations with the wait states. All the previous discussions of section 5-3.2 apply here.

2. Note that in Fig. 5-7, the $\overline{WAIT}$ signal is introduced as an active signal during TWO. The reason for this is explained in Section 5-4.1 paragraphs 3 a, b and c.

Note - During both the memory and the I/O data transfers, the $\overline{WAIT}$ line must be low during the trailing edge of the clock pulse for the CPU to go into the wait state during the next clock period.

5-5 BUS REQUEST/ACKNOWLEDGE CYCLES

1. For certain operations, the external I/O devices need to have control of the three busses from the CPU. These busses are the Address Bus (A0 - A15), the Data Bus (D0 - D7) and the Control Bus lines. In the Z-80, this mode of operation is mainly used in the DMA transfers of blocks of data between the I/O device and the memory on a cycle-stealing basis.

2. The external I/O device initiates the control of the busses by sending a signal on the $\overline{BUSRQ}$ line. The Z-80/CPU is designed such that it samples this line at the leading edge rise time of the clock pulse during the last clock period T of every machine cycle, M.

3. If the $\overline{BUSRQ}$ is active, i.e. in the low state, then at the leading edge of the next clock pulse, the CPU relinquishes control of these three busses by putting them in the high-impedance states. This is shown in Fig. 5-8. The external device then gets control of these busses and can retain this control for as many clock pulses as necessary.

4. If dynamic memory chips are used in the system and if the DMA transfers involve large blocks of data, requiring very long DMA cycles, then the external DMA controller or chip must perform the memory refresh function since the CPU does not have any direct control over the memories during DMA cycles.

5. Also note that during the bus request cycles, both the $\overline{INT}$ (interrupt request signal) and the NMI (non-maskable interrupt request) are deactivated. Neither of these two signals will be able to interrupt the CPU during bus request cycles. (See Table 1-3.)

6. The I/O device terminates this cycle by deactivating the $\overline{BUSRQ}$ signal (i.e. returning it to the high state). At the leading edge of the next clock pulse,

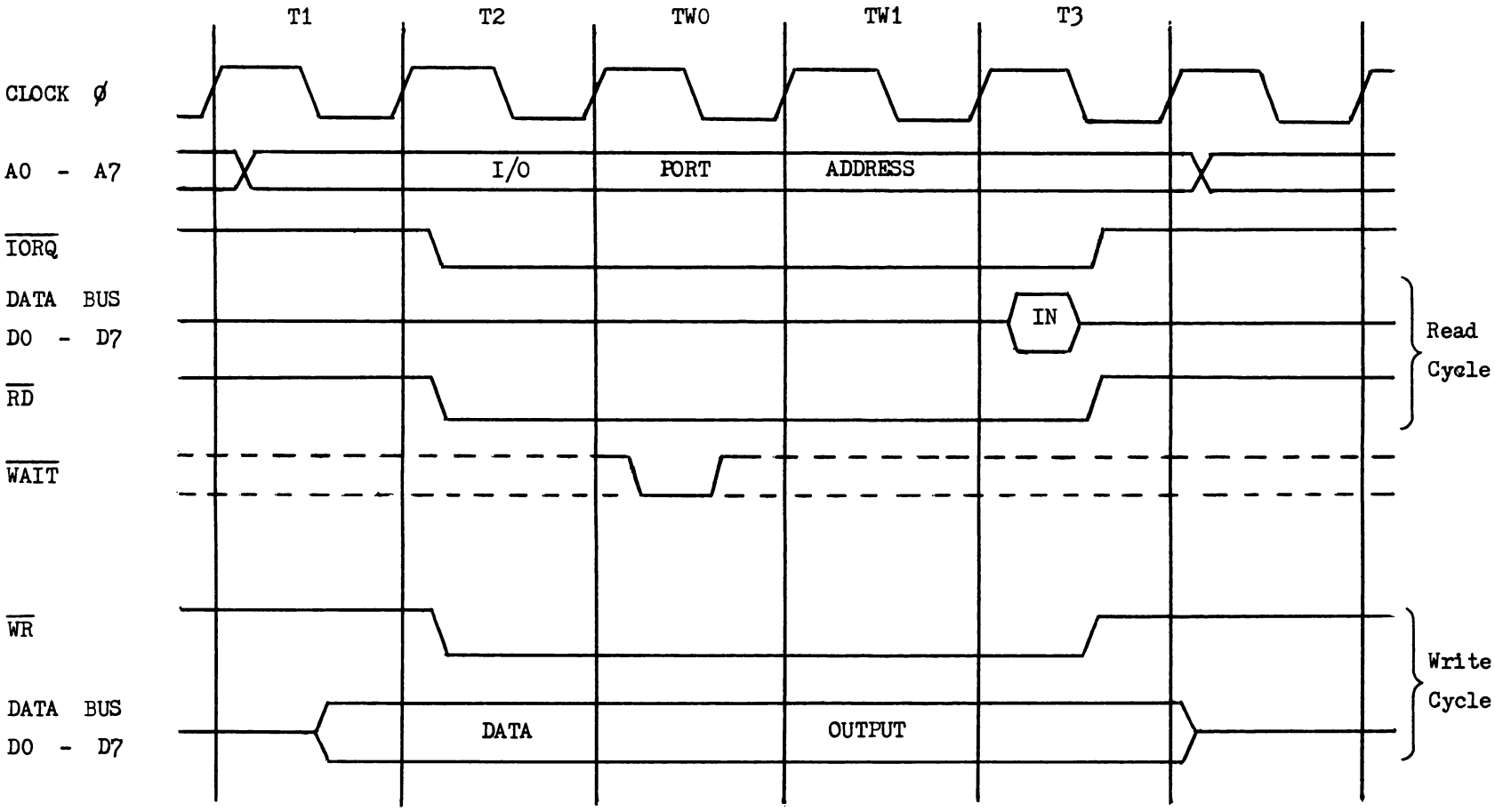


Fig. 5-7 I/O read and write cycle timing with WAIT states.

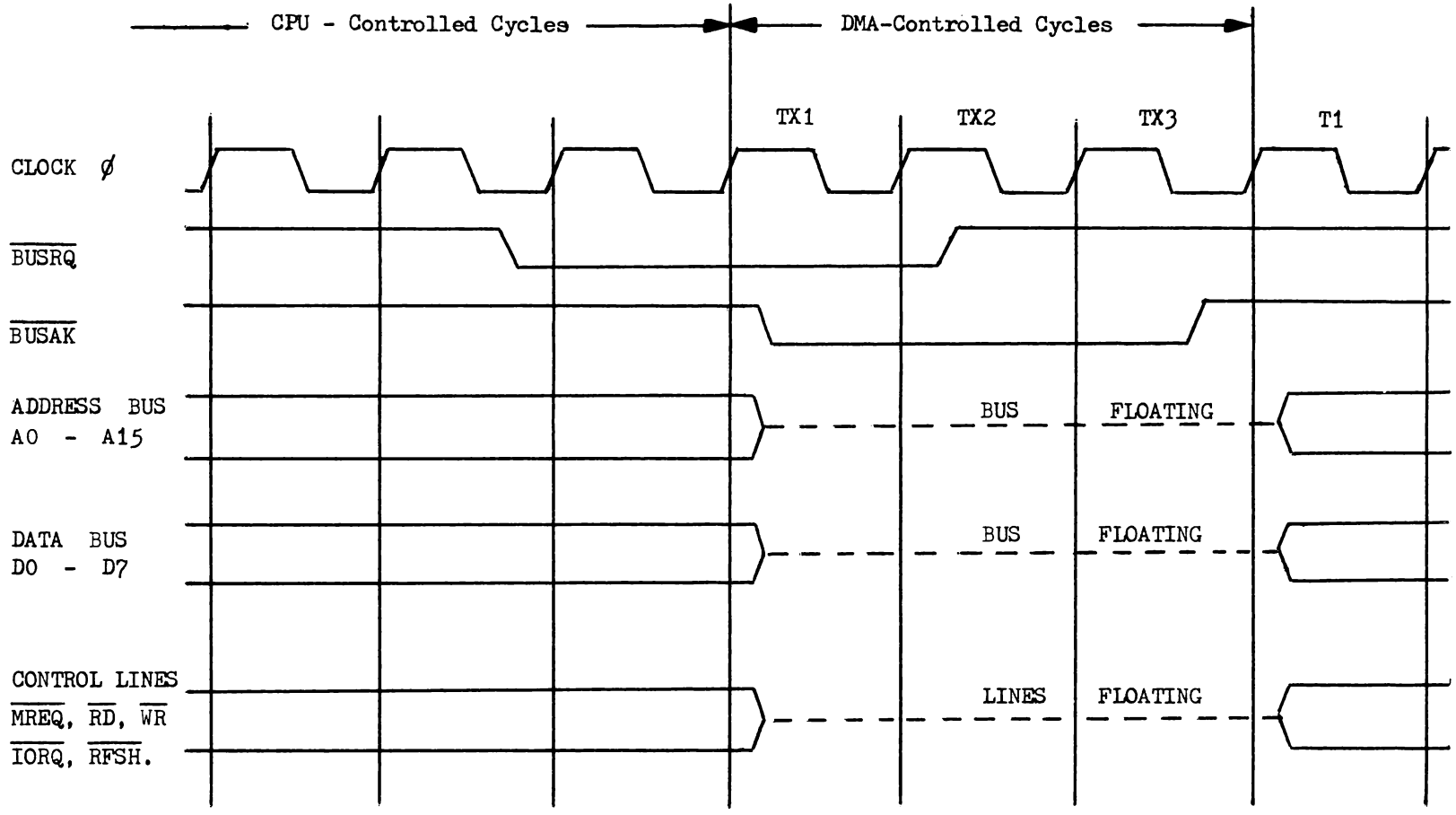


Fig. 5-8 Bus request/acknowledge cycle.

control of these busses is returned to the CPU.

5-6 MASKABLE INTERRUPT REQUEST/ACKNOWLEDGE CYCLE

1. Figure 5-9 shows the timing involved in the maskable interrupt cycle. A maskable interrupt refers to the situation where the interrupt can be masked or overriden by means of CPU software. The timing chart of Fig. 5-9 assumes that this feature is not activated under program control and so the CPU will honor this incoming interrupt.

2. At the leading edge of the last clock pulse of all instructions, the CPU samples the INT line. If it is low, then the CPU goes into a special M1 cycle starting with the leading edge of the next clock pulse.

3. As shown in Fig. 5-9, the M1 cycle consists of 5 T states with two special wait states automatically inserted between T2 and T3. As mentioned in Section 1-3, a daisy chain priority interrupt scheme is available in the Z-80/CPU. Since a daisy chain involves rippling a signal through the logic hardware associated with each I/O device, to identify the interrupting I/O, this signal requires time. This additional time is provided by TW1 and TW2. This allows ample time for the interrupting device to place its unique 8-bit vector or address on the Data Bus.

4. Note that the $\overline{M1}$ line is active during the M1 cycle, and when $\overline{IORQ}$ is simultaneously active (during the second half of TW1 and TW2), it indicates that the interrupt is acknowledged and the interrupting device's identifying vector can be placed on the Data Bus.

5. During the M1 cycle, the $\overline{MREQ}$ signal is inactive.

5-7 NON-MASKABLE INTERRUPT REQUEST ACKNOWLEDGE CYCLE

1. This interrupt has a higher priority over the INT signal and can override it. It cannot be disabled under software control. The principal use of this interrupt is to indicate an impending power failure to the CPU, which immediately responds to it by taking the following action:

   a. The response is very similar to a normal memory read operation, except that the CPU ignores the information on the Data Bus.

   b. The contents of the PC are automatically pushed into the Stack.

   c. The program automatically branches to a service subroutine whose starting address is 0066_H.

2. The $\overline{NMI}$ line is sampled by the CPU at the leading edge of the last clock pulse of the last M cycle as shown in Fig. 5-10. If it is low, then the CPU starts a

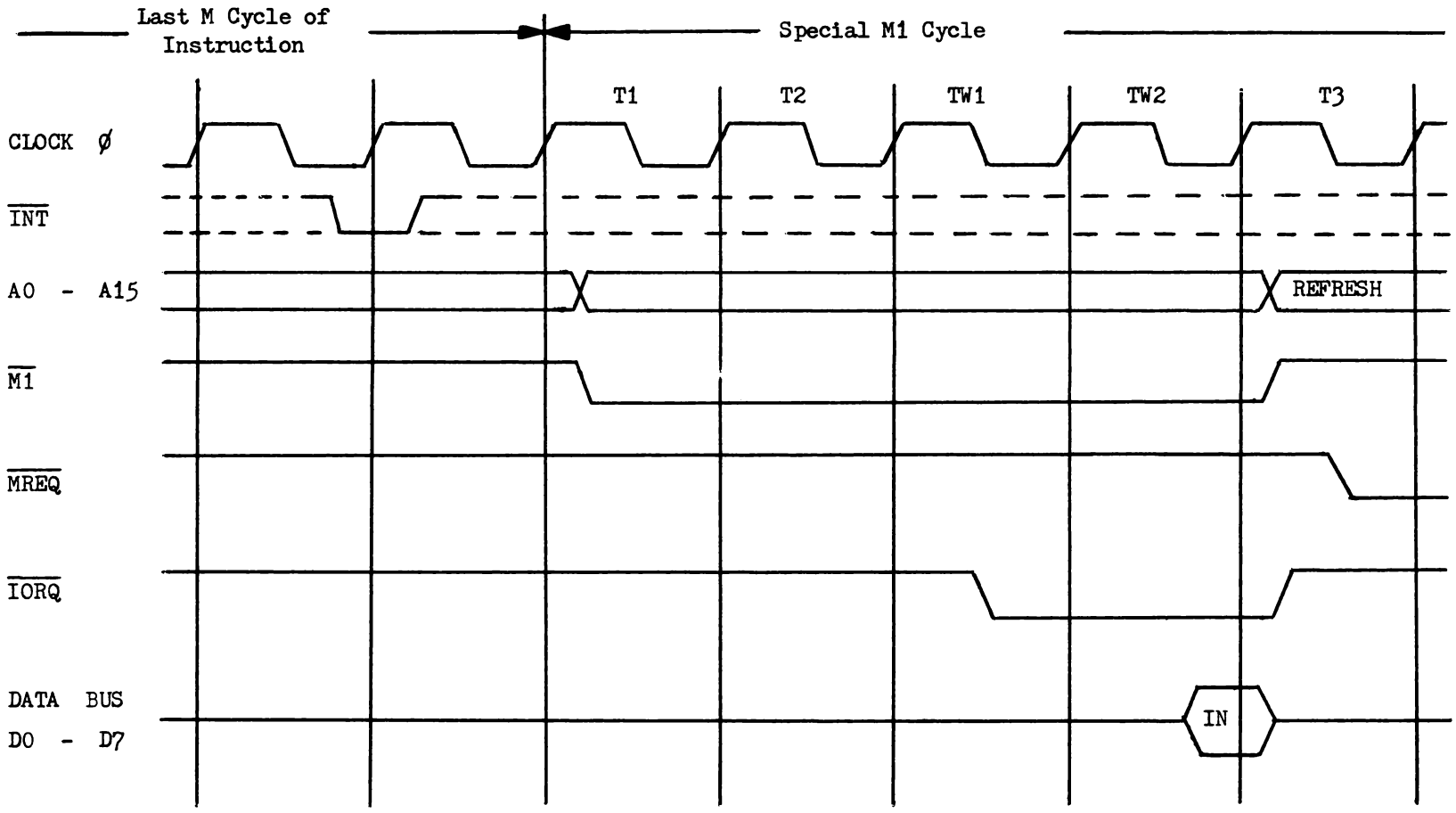


Fig. 5-9 Maskable interrupt request/acknowledge cycle.

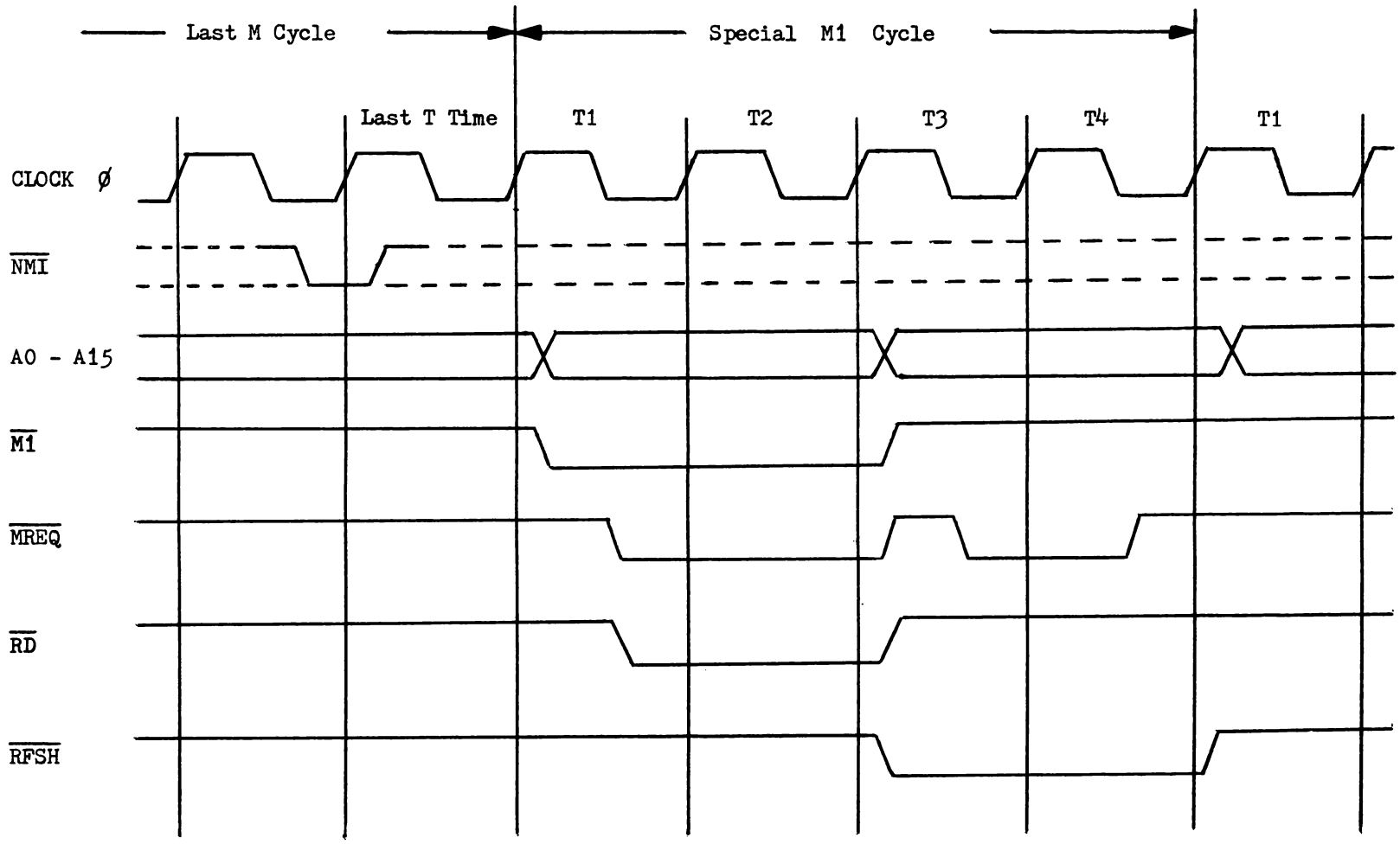


Fig. 5-10 Non-maskable interrupt/request cycle.

special M1 cycle at the leading edge of the next clock pulse.

3. This cycle consists of four T states. The operations of the various signals are similar to those which were described in previous sections and so will not be repeated here.

5-8 EXIT FROM HALT INSTRUCTION CYCLE

1. When the CPU encounters a HALT instruction in the program, it starts to execute a series of NOP instructions which in effect idle the CPU.

2. The CPU continues to idle until it receives an interrupt from an external I/O device. The interrupt could be either a maskable interrupt, $\overline{INT}$, or a non-maskable interrupt, $\overline{NMI}$. If an $\overline{INT}$ signal comes along, then the software interrupt disable flip-flop must be set, otherwise the CPU will not honor the $\overline{INT}$.

3. Both the $\overline{INT}$ and the $\overline{NMI}$ lines are sampled by the CPU at the leading edge of the T4 clock pulse as shown in Fig. 5-11. If either $\overline{NMI}$ or $\overline{INT}$ with the IFF enabled is received, the CPU will exit from or terminate the HALT state and resume operation at the leading edge of the next clock pulse T1. The following cycle will then be an interrupt acknowledge cycle, depending upon whether the $\overline{INT}$ or the $\overline{NMI}$ signal was received. (See Table 1-3.)

4. If both $\overline{INT}$ and $\overline{NMI}$ are simultaneously active, as shown in Fig. 5-11, then only the $\overline{NMI}$ will be acknowledged since it has a higher priority than $\overline{INT}$.

5. During the HALT state, NOP instructions are continuously executed. This is done so that the memory refresh signals are kept active for the dynamic memories while the CPU is idling. Otherwise the data in the memory chips will be destroyed.

6. Also note that the HALT cycle is executed during the M1 cycle which is normally the OP code fetch cycle. However, during the HALT cycle, the CPU ignores any data placed by the memory on the Data Bus and the CPU is forced to execute NOP instructions.

5-9 REVIEW QUESTIONS

5-1 In the Z-80/CPU, the clock period is defined as:

a. The clock frequency.
b. The T cycle.
c. The M cycle.
d. The clock Ø

5-2 In the Z-80/CPU, the clock period is the time interval between

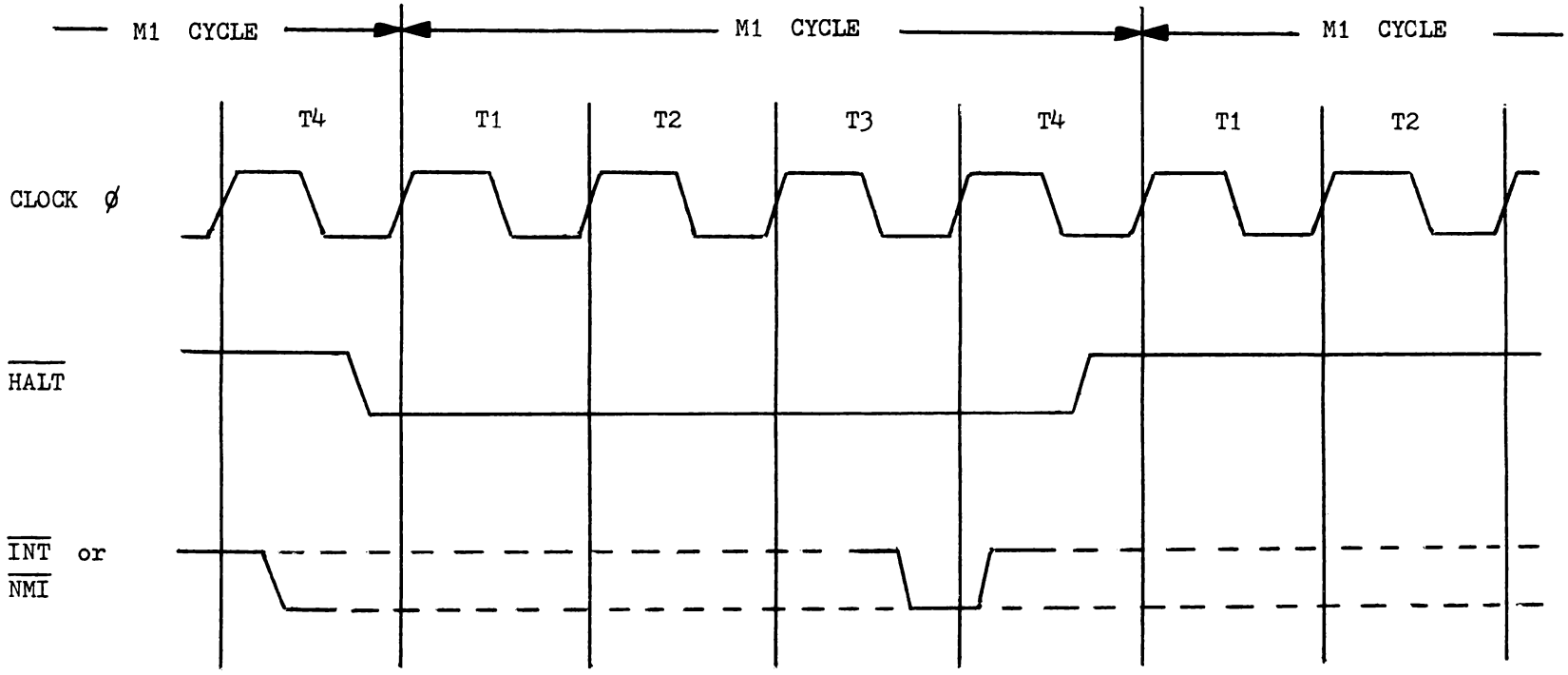


Fig. 5-11 Exit from HALT cycle.

a. The leading edge of one negative-going pulse to the leading edge of the next negative-going pulse.

b. The trailing edge of one positive-going pulse to the trailing edge of the next positive-going pulse.

c. The trailing edge of one positive-going pulse to the leading edge of the next negative-going pulse.

✓ d. The leading edge of one positive-going pulse to the leading edge of the next positive-going pulse.

5-3 Indicate true or false for the Z-80/CPU.

a. ___ Each basic operation normally takes 3-6 clock periods to complete.

b. ___ The time required to perform the basic operations cannot be extended beyond 6 clock periods.

c. ___ More clock periods are required to synchronize the slower peripherals with the faster CPU.

d. ___ The instruction cycle is broken down into subcycles called the M cycles.

5-4 Answer yes or no for the Z-80 instruction cycle.

a. ___ Do all instructions contain the same number of M cycles?

b. ___ Can the M cycles contain different numbers of T cycles?

c. ___ Is it possible to lengthen the M1 cycle?

5-5 Indicate true or false for the normal instruction cycle.

a. ___ Normally, M1 consists of 3-5 T cycles.

b. ___ M1 can be lengthened by a WAIT signal.

c. ___ M2 and M3 are normally 4-6 T cycles long.

d. ___ M2 and M3 cannot be lengthened by a WAIT signal.

e. ___ M2 and M3 transfer data between I/O and CPU.

5-6 Fill in the blanks for the OP code fetch cycle.

a. The M1 cycle contains _____ T cycles. (3; 4; 5; 6)

b. The first step transfers the contents of the _____ to the memory. (ALU; CPU; PC; I/O)

c. The address is activated during the _____ clock periods. (T1 and T2; T2 and T3; T1 and T3; T3 and T4)

d. During T3 and T4, the _____ lines of the address bus send the memory refresh address. (Upper 8; lower 7; lower 8; upper 7)

5-7 During an OP code fetch cycle, which signal is sent by the CPU to the memory to inform that the bits on the Address Bus are valid address bits?

a. $\overline{\text{IORQ}}$
b. $\overline{\text{RD}}$
c. $\overline{\text{MREQ}}$
d. $\overline{\text{RFSH}}$

5-8 Which signal from the CPU informs the memory that the OP code fetch is a memory read operation?

a. $\overline{\text{RFSH}}$
b. $\overline{\text{RD}}$
c. $\overline{\text{WAIT}}$
d. $\overline{\text{MREQ}}$

5-9 Refer to Fig. 5-2. The $\overline{\text{MREQ}}$ signal is active (i.e. goes low) during the second half of T1 because,

a. Of an error in the logic design of the CPU.
b. The $\overline{\text{M1}}$ signal is activated during the first half of T1.
c. The $\overline{\text{RD}}$ signal is activated during the second half of T1.
✓ d. It gives the address bus ample time to stabilize during the first half of T1.

5-10 Refer to Fig. 5-2. Indicate true or false for the $\overline{\text{M1}}$ signal.

a. ___ It deactivates the Address Bus A0-A15.

b. ___ It activates the Data Bus D0 - D7.

c. ___ It does not signal a write operation to the memory chips.

d. ___ It indicates that the current cycle is an OP code fetch cycle.

5-11 Refer to Fig. 5-2 and answer yes or no.

a. ___ Is it possible to generate $\overline{\text{M1}}$ more than once?

b. ___ Is it possible to have more than one byte in the OP code?

c. ___ Is it possible to place the memory refresh address on the Address Bus, A0 - A15, during $\overline{\text{M1}}$ of the OP code fetch cycle?

5-12 During the OP code fetch cycle, the refresh address is stable on the Address Bus when

a. $\overline{\text{MREQ}}$ is active
b. $\overline{\text{MREQ}}$ is inactive
c. $\overline{\text{RD}}$ is active
d. M1 is active

5-13 Indicate true or false for the OP code fetch operation with WAIT states. (Fig. 5-3)

a. ___ During T2, the CPU samples $\overline{\text{WAIT}}$ line.

b. ___ If the $\overline{\text{WAIT}}$ line is high, the CPU then goes into a wait state for the next T period.

c. ___ When $\overline{\text{WAIT}}$ is low, all CPU signals are maintained in their respective positions as at the end of T1.

d. ___ During every subsequent TW cycle, the CPU samples the $\overline{\text{WAIT}}$ line.

e. ___ T3 and T4 are resumed only when $\overline{\text{WAIT}}$ is high during the last TW period.

5-14 The memory read cycle requires _____ clock periods and the memory write cycle requires _____ clock periods. (2; 2 or 3; 3; 3 to 5; 2 to 4).

5-15 The following statements relate to the memory read/write cycles without $\overline{\text{WAIT}}$ states. Indicate true or false.

a. ___ $\overline{\text{MREQ}}$ is active during T1 and T2 but inactive during T3.

b. ___ $\overline{\text{MREQ}}$ can be directly used as a chip enable signal in dynamic memory chips.

c. ___ The $\overline{\text{WR}}$ signal informs the memory that the CPU has placed valid data on the Data Bus.

d. ___ It is possible to use $\overline{\text{WR}}$ as R/$\overline{\text{W}}$ signal for almost any semiconductor chip.

5-16 Refer to Fig. 5-5. During read operation, data are placed on the Data Bus only during T3. For write operation data are placed on the Data Bus during second half of T1, T2, through the wait states and T3. The reason for this is that,

a. The Data Bus is not busy with other things during the write operations.
b. Address Bus is tied up with the OP code's address during the read memory operation.
c. It assures sufficient time for the memory to write into the addressed location.
d. The $\overline{\text{WR}}$ line is high during this time frame.

5-17 The following statements relate to the I/O read/write cycles without WAIT states. (Fig. 5-6) Indicate true or false.

a. ___ The $\overline{\text{IORQ}}$ signals that the bits on the Address Bus are valid address bits.

b. ___ The $\overline{\text{IORQ}}$ indicates valid I/O address on the lower 8-bit lines of the Address Bus.

c. ___ TWO cycle is intentionally introduced in the I/O read/write cycle.

d. ___ The TWO cycle helps with the speed incompatibility problem when slower speed MOS technology is used for I/O devices.

5-18 During both the memory and the I/O data transfers, the $\overline{\text{WAIT}}$ line must go low during the _____ edge of the clock pulse for the CPU to go into the wait state during the next clock period. (Leading; trailing)

5-19 The following statements relate to the Bus Request/Acknowledge cycles. Indicate true or false.

a. ___ The $\overline{\text{BUSRQ}}$ signal is sent by the CPU.

b. ___ The CPU samples the $\overline{\text{BUSRQ}}$ line at the leading edge of rise time of the clock pulse during the last T period of every M cycle.

c. ___ If $\overline{\text{BUSRQ}}$ is active, the CPU relinquishes control of the busses immediately in the same T cycle.

d. ___ During Bus Request cycles, both $\overline{\text{INT}}$ and $\overline{\text{NMI}}$ are deactivated.

e. ___ The bus request cycle is terminated by the I/O device by putting the $\overline{\text{BUSRQ}}$ signal in the high state.

5-20 A maskable interrupt refers to the condition where the interrupt _____ be overriden by means of the CPU software. (Can; cannot)

5-21 Refer to Fig. 5-9 for the maskable interrupt request/acknowledge cycle and indicate true or false.

a. ___ The CPU samples the $\overline{\text{INT}}$ line at the leading edge of the last clock pulse of all instructions.

b. ___ If $\overline{\text{INT}}$ is low, the CPU goes into the special M1 cycle starting with the trailing edge of the next clock pulse.

c. ___ Wait states TW1 and TW2 are automatically inserted during bus request cycles.

d. ___ TW1 and TW2 ensures adequate time for the interrupting device to place its 8-bit vector on the Data Bus.

5-22 The _____ signal indicates that the interrupt is acknowledged and that the interrupting device can place its identifying vector on the Data Bus. ($\overline{\text{IORQ}}$; $\overline{\text{MREQ}}$; $\overline{\text{INT}}$)

6 Z-80 PARALLEL INPUT OUTPUT CHIP

CHAPTER OBJECTIVES

1. To introduce students to the programmable parallel I/O chip.
2. To present the principal features of the Z-80 parallel input-output (PIO) chip.
3. To present and describe the generalized architecture of the PIO.
4. To present the various registers involved in the logic of each I/O port and describe the functions performed by each logic block.
5. To present the Interrupt Control Protocol involved in PIO operations.
6. To present the package pin assignments and functions in convenient tabular form.
7. To describe and discuss the interrupt timing associated with all three operational modes.
8. To present the timing involved in interrupt acknowledge and interrupt servicing.
9. To show how the daisy chain can be extended without using additional external logic.

TEXTBOOK REFERENCES

For reviewing material in the main hardcover textbook, relevant to the topics covered in this chapter, the following chapters and/or sections are suggested:

1. For general discussion of interrupt I/O - Chapter 9
2. For daisy chain logic - Sec. 9-7
3. For vectored-interrupt sequence of events - Sec. 9-5.2
4. For vectored-interrupt logic/operation - Sec. 9-5.3

5. For general description of programmable I/O interfaces - Chapter 12

6. For features and capabilities of programmable parallel interfaces - Secs. 12-3.1 and 12-3.2.

6-1 INTRODUCTION

The Parallel Input-Output (PIO) chip provides a programmable, 2-port, interface between parallel peripheral I/O devices and the Z-80/CPU. Under program control, the CPU is capable of reconfiguring the PIO so that it can interface with a wide variety of peripherals such as paper tape readers, most keyboards, printers, etc. without requiring additional external logic. Data transfers are accomplished by interrupt-controlled techniques which allow the external I/O devices to initiate such transfers. An interrupt priority system is provided using the daisy chain technique.

6-2 PRINCIPAL FEATURES OF THE PIO

1. The PIO is a third generation chip which uses the N-channel, depletion-mode, silicon-gate technology.

2. A single, +5 $\pm$5%, power supply is required.

3. A single-phase, TTL level, clock is used.

4. The PIO chip operates in a temperature environment of 0°C to +70°C.

5. On the peripheral side, all inputs as well as the outputs are TTL compatible.

6. Two I/O ports are provided and they are completely independent of each other.

7. There are four distinct and separate modes of port operations. Both ports, A and B, have the following three modes (a, b and c):

   a. Byte output - unidirectional (Mode 0)
   b. Byte input - unidirectional (Mode 1)
   c. Individual bits control - input or output (Mode 3)
   d. Additionally, Port A has a bidirectional byte control mode. (Mode 2)

8. Both ports have interrupt-driven handshake capability (i.e. status control signals).

6-3 THE PIO ARCHITECTURE

6-3.1 The Generalized PIO Block Diagram

1. Figure 6-1 shows a generalized block diagram of the PIO chip.

2. On the computer side, the chip provides I/O interface logic for transferring data on an 8-bit data bus Do - D7 and a 6-bit control bus is also used.

3. The Port A I/O and the Port B I/O logic each communicates with the external I/O device via an 8-bit data bus, respectively A0 - A7 and B0 - B7. These lines could be either unidirectional or bidirectional depending upon the selected operational mode.

4. Each port also has 2 handshake lines to their respective I/O devices.

5. Also on the computer side, the PIO communicates with the CPU via 3 interrupt control lines, all of which are unidirectional.

6. The primary function of the Internal Control Logic Block is to synchronize the CPU data bus to the interfaces involved in each I/O port. In other words, synchronize the data flow between the CPU and the I/O devices.

7. The Interrupt Control Logic primarily deals with interrupt operations and the daisy chain priorities.

6-3.2 The Port Logic Block Diagram

1. Figure 6-2 is a block diagram of the block labelled Port A or Port B in the generalized PIO block diagram of Fig. 6-1.

2. Figure 6-2 shows six registers and the logic block for the handshake control.

3. The Mode Control Register is first loaded by the CPU with a 2-bit code which selects one of the 4 operating modes mentioned in Sec. 6-2, paragraph 7. This code sets up the logic configuration for the desired PIO operating mode.

4. The 8-bit data output register receives the byte from the CPU and sends it out of the PIO chip to the I/O device via the 8-bit Data/Control bus.

5. Likewise, the 8-bit data input register receives the incoming byte from the I/O device via the 8-bit Data/Control Bus and transmits it to the CPU via the Internal Bus.

6. The actual data transfers, between the CPU and the I/O devices, are controlled by the 2 status control lines via the handshake control logic. The exact functions of these two lines are explained later.

7. The 8-bit I/O Select logic is used only in the Bit Control Mode (mode 3). This register is first loaded by the CPU with a control word. This word then assigns the 8 lines of the I/O Data/Control Bus individually as either an input or an output line. The bus then becomes a combination of designated input and output lines.

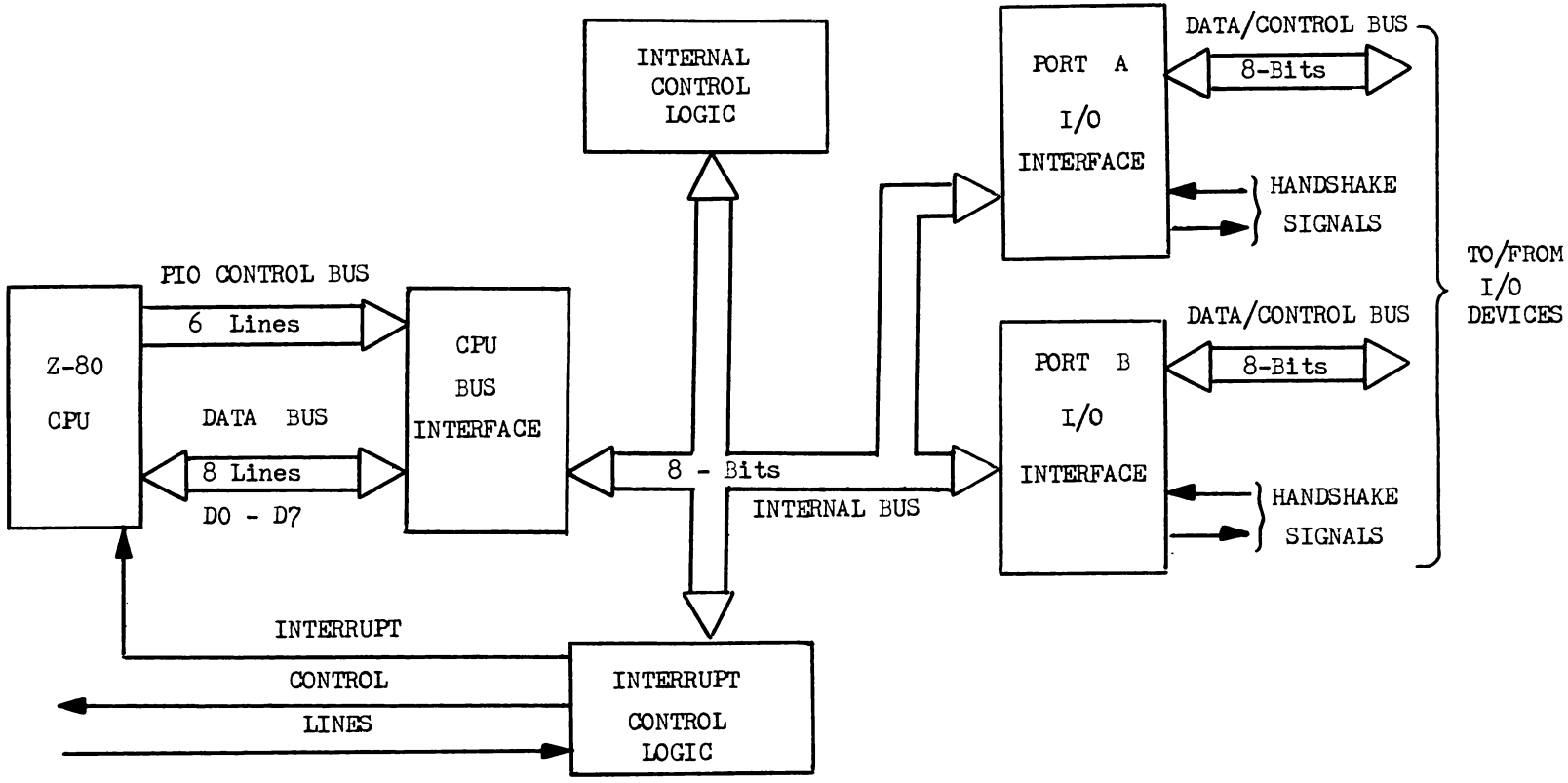


Fig. 6-1 Generalized PIO block diagram.

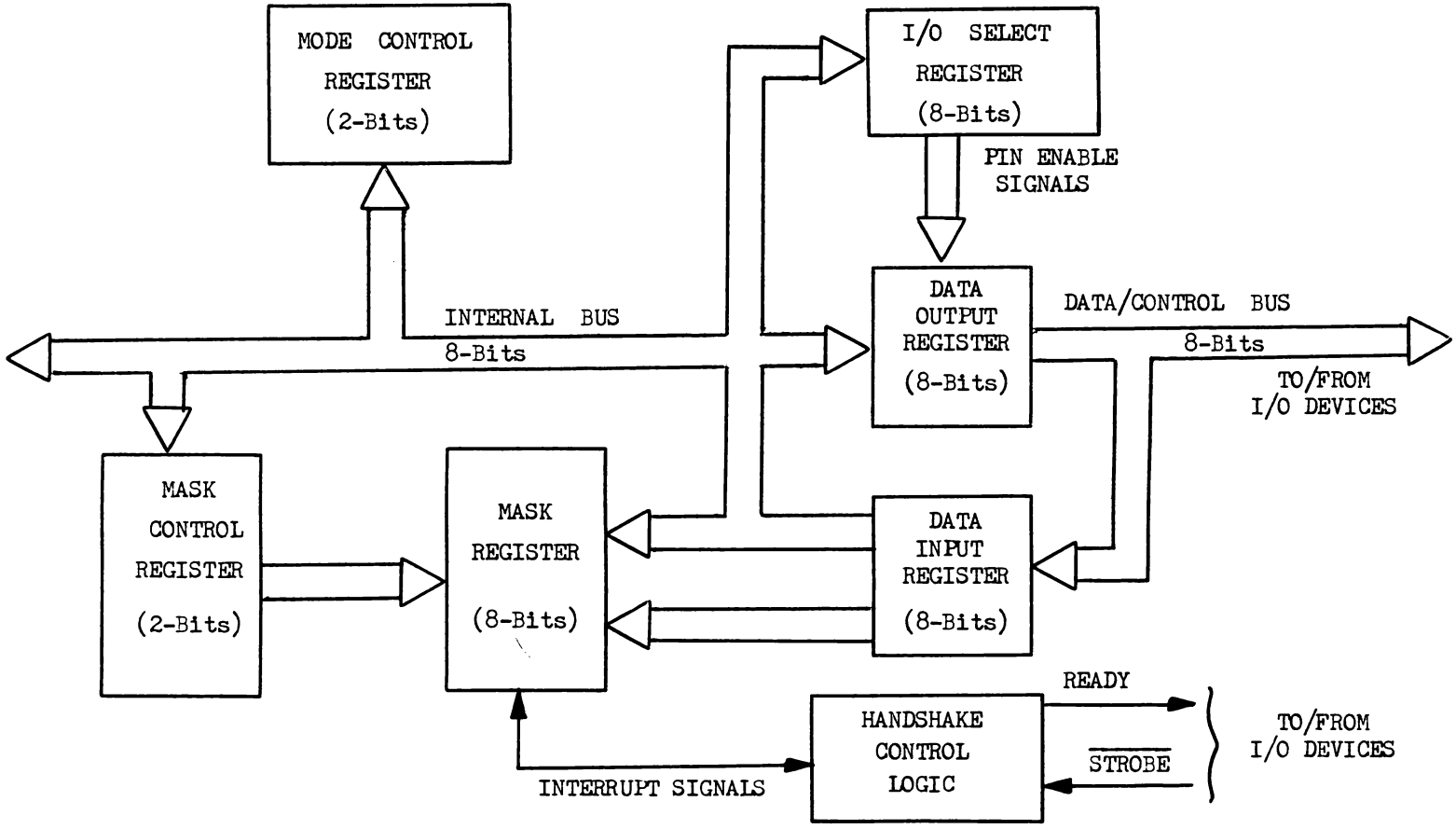


Fig. 6-2 Port Logic block diagram.

8. The 8-bit Mask Register is, likewise, used only in the Bit Control Mode (mode 3). It is used for a very special interrupt feature of the Z-80 system. Again, this register is preloaded with a word from the CPU. The word in this register effectively masks out the selected I/O pins of the I/O Data Control Bus. Then, when any or all of the unmasked pins reach a pre-specified logical state (either high or low), an interrupt is generated.

9. The 2-bit Mask Control Register, which is also preloaded by the CPU defines the desired active states (either high or low) for the unmasked bits of the Mask Register. It also performs another function. It defines whether the interrupt should be generated when all the unmasked bits of the bus are in the defined state (the AND function) or when any one or more of the unmasked bits are in the active state (the OR function).

6-3.3 The Interrupt Control Protocol

1. Since the daisy chain technique is used, the priority of each I/O device is established by its position in the chain relative to the CPU. The device closest to the CPU has the highest priority.

2. In the PIO, Port A has a higher priority than Port B.

3. When the PIO is operated in either Mode 0, 1 or 2, an interrupt request is sent to the PIO when the I/O device is ready for a data transfer.

4. When the PIO is operated in Mode 3 (bit control mode), an interrupt can be generated only when the state of the I/O device matches a programmed value.

5. In the PIO, interrupts can be nested, i.e. multi-level priority interrupt capability is available. An I/O device with a lower priority on the daisy chain cannot interrupt while the service subroutine of a device with a higher priority is being executed. However, a device with a higher priority can interrupt during the subroutine execution of an I/O device having a lower priority on the daisy chain.

6. When operating in Mode 2 (bidirectional byte transfer mode), the interrupting I/O sends an 8-bit vector (i.e., its own unique address) to the CPU. This vector forms a pointer. This points to a location in the memory where the starting address of the interrupt service subroutine is stored. (see Sec. 5-4.3.)

6-4 PACKAGE PIN ASSIGNMENTS AND FUNCTIONS

The PIO is contained in a standard 40-pin, dual-in-line package (DIP). The various pins and their respective functions, along with their designations are presented in convenient tabular form. Functionally, the pins can be divided into six groups as shown in Fig. 6-3. Note that some lines are shown as unidirectional and others as

bidirectional. In the latter group, some of the lines may indicate signal flow in one direction and others in the other direction. The function tables, Tables 6-1 through 6-5, clarify these with the following arrow symbols:

PIO ⟶ Indicates signals flowing from the PIO
PIO ⟵ Indicates signals flowing into the PIO
PIO ⟷ Indicates bidirectional flow on the same line(s)

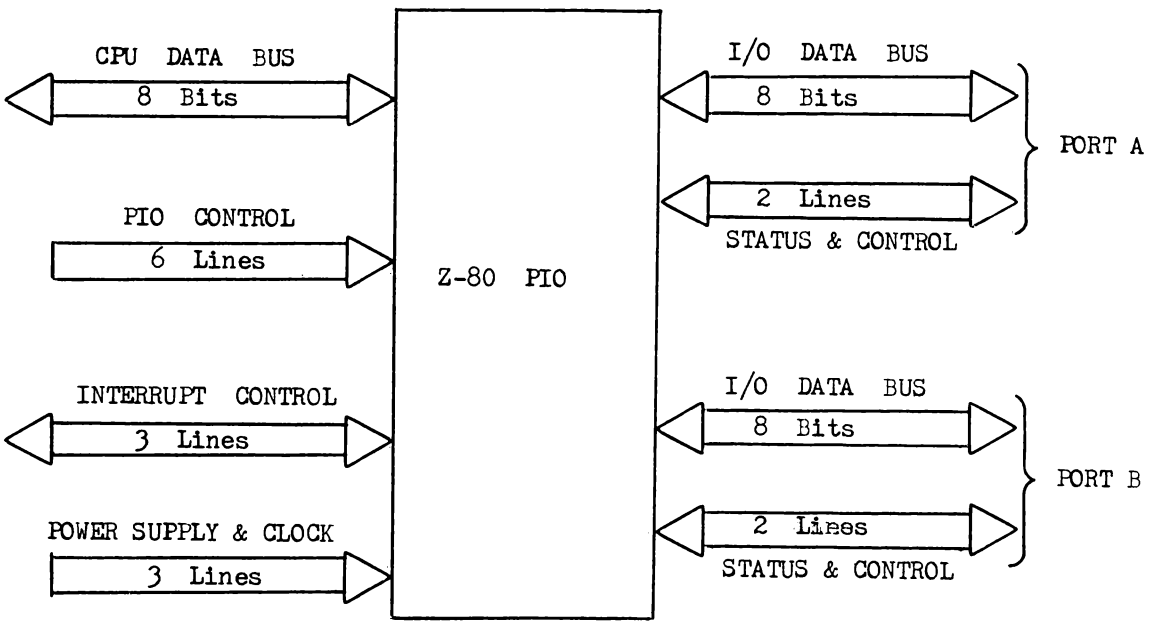


Fig. 6-3 Block diagram of CPU package pins groups.

PIN DESIGNATIONS	SIGNAL FLOW DIRECTIONS	FUNCTIONS
D0 - D7	PIO ◄——►	8-bit, tristate, CPU data bus, is active high. It is used for data and commands transfers betwwen the PIO and the CPU.
A0 - A7	PIO ◄——►	8-bit, tristate, bus for transfers of data and/or status and control signals between the I/O device and Port A.
B0 - B7	PIO ◄——►	8-bit, tristate, bus for transfer of data and/or status and control signals between the I/O device and Port B of PIO.

Table 6-2 PIO Control group.

PIN DESIGNATIONS	SIGNAL FLOW DIRECTIONS	FUNCTIONS
B/$\overline{A}$ Sel	PIO ◄——	Port select signal defines the port to be selected during a data transfer operation. A low signal selects Port A and a high selects Port B. (Note: sometimes address bit A0 from the CPU is used for this selction).
C/$\overline{D}$ Sel	PIO ◄——	Control or Data Select signal, defines whether data or control information is transferred between the CPU and the port selected by the B/$\overline{A}$ Sel signal. A low signal indicates that data bits are transferred between the CPU and the PIO. A high during CPU write indicates that a command or control word is sent to the port selected by the B/$\overline{A}$ Sel signal. (Note: sometimes address bit A1 from the CPU is used for this functions).
$\overline{CE}$	PIO ◄——	An active low signal that informs the PIO to transmit the data word to the CPU during the CPU read cycle. During the CPU write cycle, this signal informs PIO to accept either the control or the data word from the CPU as defined by the C/$\overline{D}$ Sel signal.
$\overline{M1}$	PIO ◄——	Machine cycle one signal, active low, is a SYNC pulse from the CPU which synchronizes the PIO interrupt logic. When M1 and $\overline{RD}$ are simultaneously active, the CPU is fetching an instruction from the memory. When M1 and $\overline{IORQ}$ are simultaneouslк active the CPU is acknowledging an interrupt. When M1 is active without the other two being active, the PIO goes into a reset state.
$\overline{IORQ}$	PIO ◄——	I/O request signal, active low, is used for transferring commands and data between the CPU and the PIO. When $\overline{CE}$, $\overline{RD}$ and $\overline{IORQ}$ are active, the addressed port sends data to the CPU. When $\overline{CE}$ (Continued)

Table 6-2 (Continued) PIO control group.

PIN DESIGNATIONAS	SIGNAL FLOW DIRECTIONS	FUNCTIONS
$\overline{IORQ}$ (Continued)	PIO ←	and $\overline{IORQ}$ are active and $\overline{RD}$ is not active, then the CPU writes either a data or a command word into the addressed port of the PIO. If $\overline{IORQ}$ and $\overline{M1}$ are active, the CPU acknowledges an interrupt and the port involved will place its interrupt vector on the CPU data bus, depending upon its priority in the daisy chain.
$\overline{RD}$	PIO ←	Read cycle status, active low, signal indicates that the CPU is reading a word from either the I/O or the memory. When used in conjunction with the $\overline{CE}$, $\overline{IORQ}$, B/$\overline{A}$ Sel and C/$\overline{D}$ Sel signals, the $\overline{RD}$ transfers data from the PIO to the CPU.

Table 6-3 Interrupt control group.

PIN DESIGNATIONS	SIGNAL FLOW DIRECTIONS	FUNCTIONS
$\overline{INT}$	PIO ⟶	Interrupt response signal, active low. The PIO requests an interrupt from the CPU.
IEI	PIO ⟵	Interrupt Enable In, active high. Used in the priority interrupt daisy chain. A high or active signal indicates that no other I/O device with higher priority is being currently serviced.
IEO	PIO ⟶	Interrupt Enable Out, active high. Also used in the priority interrupt daisy chain. It is active only when IEI is active and the CPU is not servicing an interrupt from the PIO. When active, it blocks interrupt signals from lower priority I/O devices from interrupting while it is being serviced.

Table 6-4 Port status and control group.

PIN DESIGNATIONS	SIGNAL FLOW DIRECTIONS	FUNCTIONS
$\overline{\text{A STB}}$	PIO ←	Port A strobe signal, active low, inputted to the PIO from the I/O device. Operation depends upon selected mode a follows: 1. Mode 0 - output byte mode - a positive-going, trailing edge of this pulse indicates that the I/O has received the data sent by the PIO. 2. Mode 1 - input byte mode - the I/O issues this signal indicating that the data sent to the input register of the selected PIO port. 3. Mode 2 - bidirectional mode - when active indicates that the data from the output register of Port A is put on its own bidirectional data bus. Then, the positive-going, trailing edge of this pulse indicates that the data is received by the I/O. 4. Mode 3 - control mode - this pulse is internally inhibited by the PIO.
A RDY	PIO →	Register A ready signal, active high. operation depends upon selected mode as follows: 1. Mode 0 - output byte mode - when active indicates that the Port A output register contains the data word, the data bus has stabilized ready for transfer to the I/O device. 2. Mode 1 - input byte mode - when active indicates that the input register of Port A, is empty and ready to accept the next word from the I/O device. 3. Mode 2 - bidirectional mode - when active indicates that the output register of Port A has the data word for transfer to the I/O device. However, data is not outputted to the data bus unless A STB is active. 4. Mode 3 - control mode - this signal in internally inhibited by the PIO.

Table 6-4 (Continued) Port status and control group.

PIN DESIGNATIONS	SIGNAL FLOW DIRECTIONS	FUNCTIONS
$\overline{\text{B STB}}$	PIO ◄——	Port B strobe signal, active low. Its operation is similar to that of the $\overline{\text{A STB}}$ signal except that in the Port A, mode 2 operation, this signal strobes data from the I/O device into the input register of Port A.
B RDY	PIO ——►	Register B ready signal, active high. Operation is similar to that of the A RDY signal except that in the Port A mode 2 operation, this signal is active when the input register of Port A is empty and ready to accept the next incoming data word from the I/O device.

Table 6-5 Power supply and clock group.

PIN DESIGNATIONS	SIGNAL FLOW DIRECTIONS	FUNCTIONS
ϕ	PIO ◄——	This is the single-phase, TTL level clock input.
+5 V	PIO ◄——	This is the single power supply of +5 $\pm$ 5% V that is needed for the PIO.
GND	PIO ◄——	This is the power supply ground.

Table 6-6 Pin number assignments.

PI DESIGNATIONS	PIN NUMBERS	PIN DESIGNATIONS	PIN NUMBERS
CPU Data Bus		PIO Control	
D0	19	B/$\overline{A}$ Sel	6
D1	20	C/$\overline{D}$ Sel	5
D2	1	$\overline{CE}$	4
D3	40	$\overline{M1}$	37
D4	39	$\overline{IORQ}$	36
D5	38	$\overline{RD}$	35
D6	3		
D7	2		
Port A I/O		Interrupt Control	
A0	15	$\overline{INT}$	23
A1	14	IEI	24
A2	13	IEO	22
A3	12		
A4	10		
A5	9	Power Supply and Clock	
A6	8	ø	25
A7	7	+5 V	26
A RDY	18	GND	11
$\overline{A STB}$	16		
Port B I/O			
B0	27		
B1	28		
B2	29		
B3	30		
B4	31		
B5	32		
B6	33		
B7	34		
B RDY	21		
$\overline{B STB}$	17		

6-5 THE PIO INTERRUPT TIMING

In this section we will discuss the timing involved in the four interrupt modes. In the following timing charts, notice that the clock phases are shown as perfect square waves. This is done only for convenience. Actually each clock pulse has its own rise and fall times. The rise and fall times of the other signals are shown as they should be. The clock pulses are used in these charts simply for reference purposes only and so they are shown as square or rectangular pulses.

6-5.1 Model 0 - Output Byte Mode

1. The timing associated with this mode is shown in Fig. 6-4. A byte output cycle is started when, as a result of an interrupt, the CPU executes an output instruction.

2. During this CPU I/O write operation, the PIO generates a $\overline{WR}$ signal. This is a composite signal obtained by mixing (i.e. AND-ing) four signals as shown below:

$$\text{(Write Data) } \overline{WR} = RD \times \overline{CE} \times \overline{C/D} \times \overline{IORQ}$$

3. The $\overline{WR}$ pulse latches the data outputted by the CPU on the data bus into the output register of the selected Port (A or B).

4. The positive-going trailing edge of the $\overline{WR}$ pulse then raises the Ready flag after the trailing edge of the next clock pulse (i.e. pulse #4 in Fig. 6-4). This flag then signals to the I/O device that the data word is now available to it.

5. The Ready signal remains high or active until one of the following two events takes place:

   a. A positive-going trailing edge of the $\overline{STROBE}$ signal indicates that the I/O device has accepted and obtained the data.

   b. If the READY is already active, then, the Ready signal will be forced into the low state 1½ clock pulses after the leading edge of the $\overline{IORQ}$ signal if the word is written into the selected port's output register.

6. The READY signal will go high on the first trailing edge of the clock pulse after the trailing edge of $\overline{IORQ}$. This then assures that the READY signal is low when the data in the selected port is in the process of changing.

7. Notice that the READY signal will not go into the low or inactive state until the negative-going edge of the clock pulse happens. This delaying of the READY signal allows a simple scheme for generating the STROBE signal. (This is accomplished by simply connecting the READY line to the STROBE line without requiring any additional external logic.)

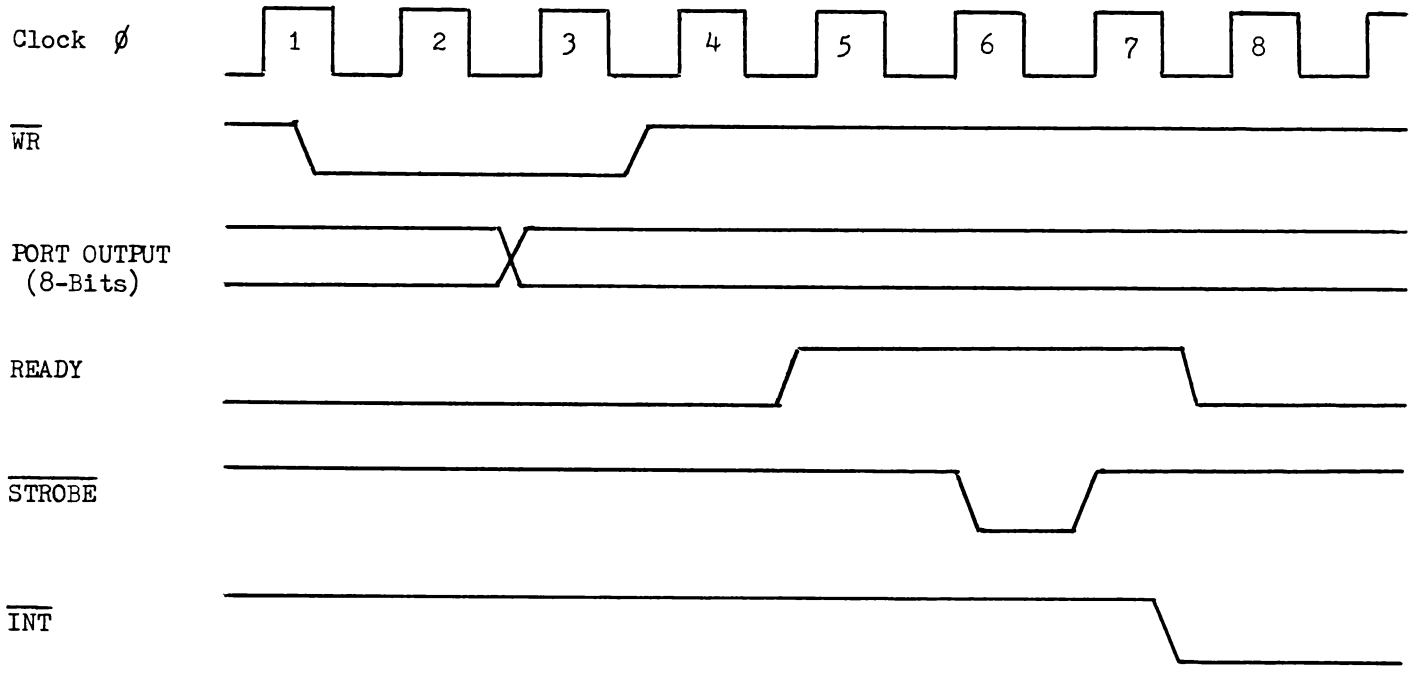


Fig. 6-4 Mode 0, byte output timing.

8. The positive-going trailing edge of the STROBE signal automatically generates an $\overline{INT}$ request provided the interrupt enable flip-flop has been set and the interrupting I/O device has the highest priority.

6-5.2 Mode 1 - Input Byte Mode

1. The timing for this mode is shown in Fig. 6-5.

2. The input cycle is initiated by the I/O device by means of a negative-going pulse on the $\overline{STROBE}$ line. This pulse loads the data word into the input register of the selected port.

3. The positive-going trailing edge of the $\overline{STROBE}$ pulse activates the $\overline{INT}$ signal provided the interrupt enable flip-flop is set and the interrupting I/O device has the highest priority.

4. The trailing edge of the next clock pulse (pulse #3 in Fig. 6-5) puts the READY line in the low or inactive state. This signifies that the input register in the PIO contains a data word and further loading of this register must be prevented.

5. The CPU executes the service subroutine and reads in the data from the selected I/O port in the PIO.

6. Upon completion of this operation, the positive-going, trailing edge of the $\overline{RD}$ signal (from the CPU) raises the READY signal to the high level after the negative-going transition of the next clock pulse (pulse #11 in Fig. 6-5). This indicates to the I/O device that the next word can be loaded into the input register of the PIO.

7. If the READY line is already active, then this line will be forced into the low state 1½ clock periods following the leading edge of the $\overline{IORQ}$ signal.

8. Note that the $\overline{RD}$ is a composite signal obtained by mixing the following four signals:

$$\text{(Read Data)}\quad \overline{RD} = \overline{RD} \times \overline{CE} \times \overline{C/D} \times \overline{IORQ}$$

6-5.3 Model 2 - Bidirectional Mode

1. Figure 6-6 shows the timing of this mode which is a combination of Modes 0 and 1. It utilizes all 4 handshake lines and is available for only Port A. When Port A is operated in Mode 2, Port B must be set to Mode 3.

2. The timing for Mode 2 is almost identical to the timing for Mode 0 (Fig. 6-4) and Mode 1 (Fig. 6-5). In Mode 2 operation, the handshake lines of Port A are used for data input control.

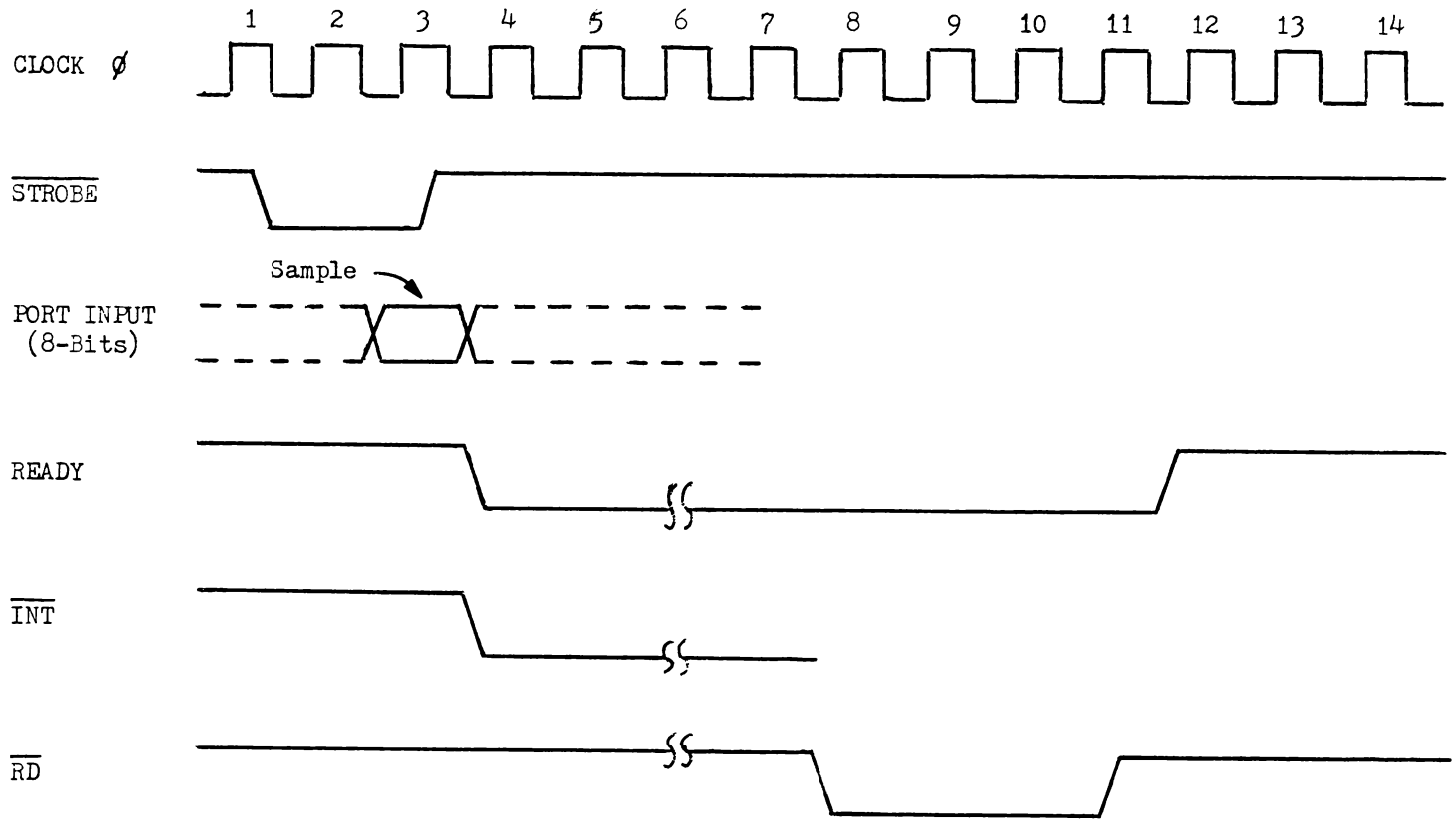


Fig. 6-5 Mode 1, byte input timing.

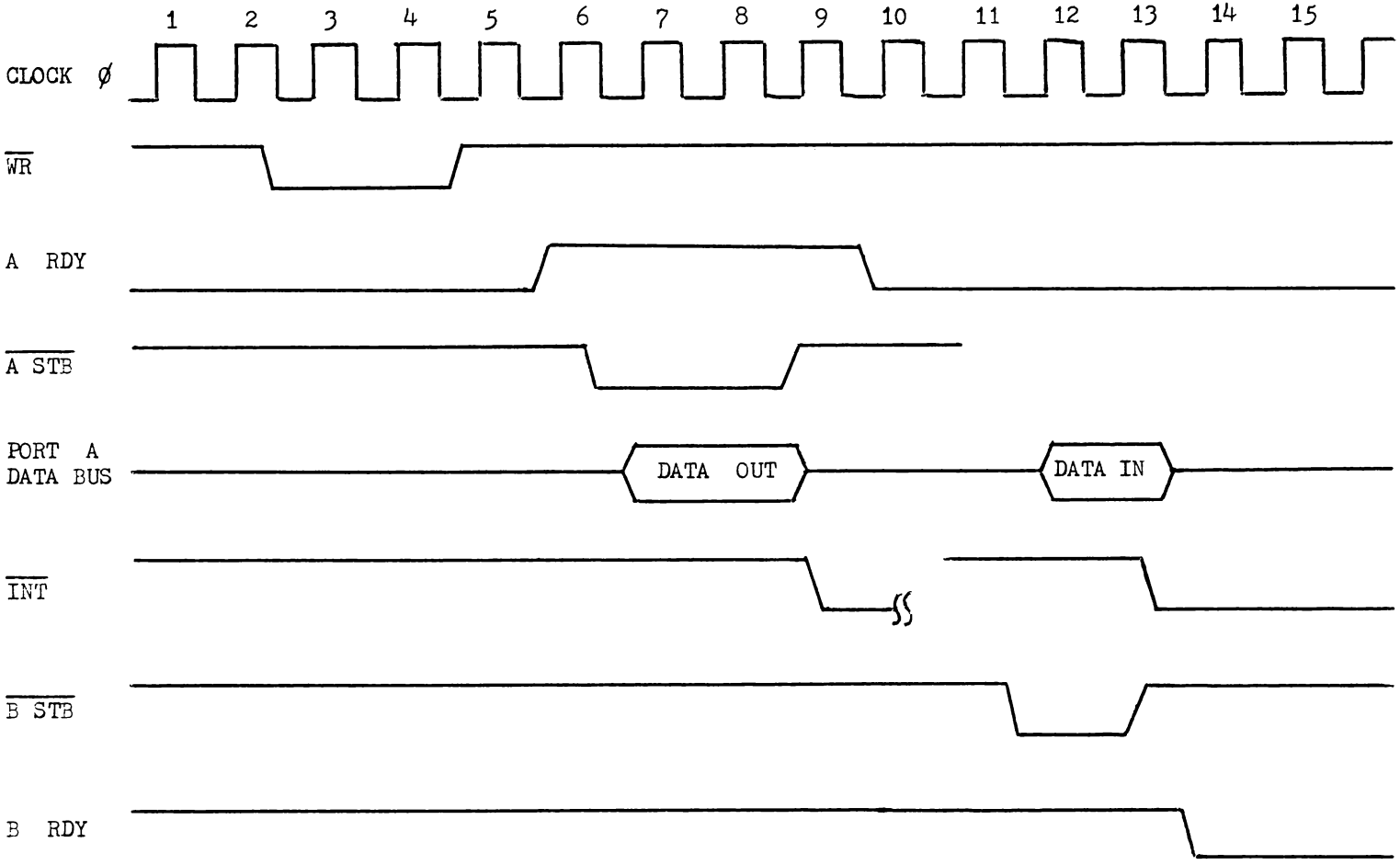


Fig. 6-6 Mode 2, Port A, bidirectional byte transfer timing.

3. In Mode 2 operation, the data word is gated out to the bus only when the $\overline{\text{A STB}}$ is active, i.e. low. The positive-going, trailing edge of this signal latches the data word into the I/O device.

4. In Mode 2, the data input operates in an identical manner to the Mode 1 operation. Also, in Mode 2, both Parts A and B must have their interrupt enables set so that bidirectional transfers are truly interrupt-driven.

5. The data inputted into the PIO are latched into the input register when the $\overline{\text{B STB}}$ is active or low. This avoids the possibility of the data being gated on to the bus, by the I/O device, when the A RDY is active or high.

6. The $\overline{\text{WR}}$ is a composite of the following signals:

$$\overline{\text{WR}} = \text{RD} \times \overline{\text{CE}} \times \overline{\text{C/D}} \times \overline{\text{IORQ}}$$

6-5.4 Mode 3 - Bit Control Mode

1. In Mode 3, the handshake signals are not used so it is possible to execute a normal port read or a normal port write any time.

2. When writing into the port, the data word is latched into the output register of the selected port using the same timing as that for Mode 0, byte output operation as shown in Fig. 6-4. Mode 3 timing is shown in Fig. 6-7.

3. When Port A is operated in Mode 3, the A RDY line is forced into a low state. When Port B is operated in Mode 3, the B RDY line is held low unless when Port A is operated in Mode 2. In that case B RDY is not affected at all.

4. During the PIO read operation in Mode 3, the data or information, transmitted to the CPU by the PIO will be composed of the following two segments:

   a. Bits from the output register whose lines are designated as output lines, and
   b. Bits from the input register whose lines are designated as input lines.

5. Note that the input register will contain the word which was in it immediately before the negative-going edge of the $\overline{\text{RD}}$ signal.

6. Remember that an interrupt will be generated only if,

   a. The interrupts from the port are enabled, and
   b. If the word on the port data line conforms to the logical conditions that were preloaded into the 8-bit Mask register and the 2-bit Mask Control register of the PIO.

7. Another interrupt signal will not be generated by the PIO until after a change in the above logical conditon occurs.

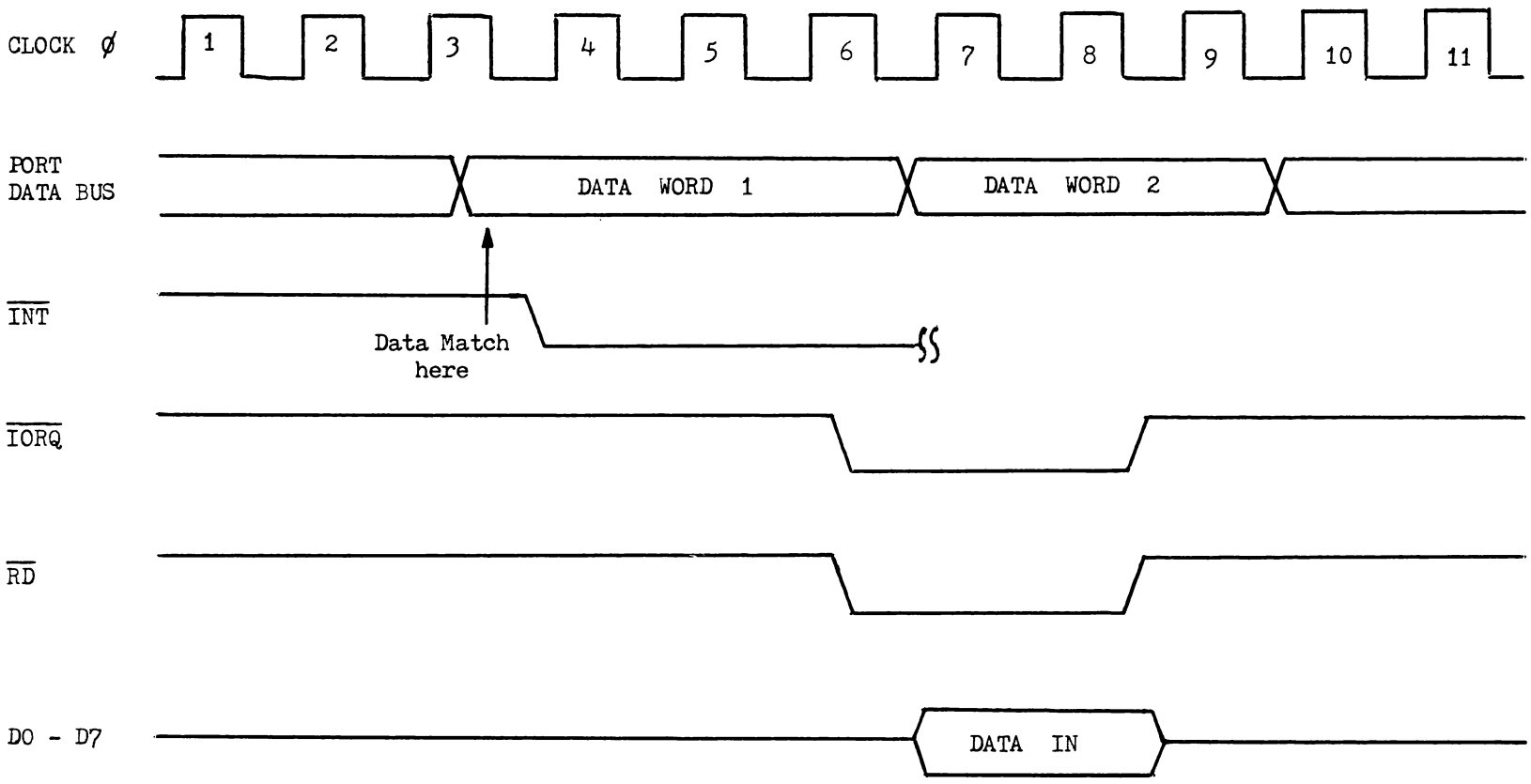


Fig. 6-7 Mode 3, bit control mode timing.

6-5.5 Interrupt Acknowledge and Servicing

In the Z-80 system, it is normally possible to form a daisy chain by connecting up to four PIO chips in tandem or in series without the need for any additional external logic. Since two I/O devices can be connected to each PIO chip, this means that up to 8 I/O devices can be normally used on the daisy chain. Later we will see how the daisy chain can be extended even further. However, in order to understand the limitations and how to overcome them we will first see how the Interrupt Acknowledge works and how an interrupt is serviced.

1. Figure 6-8 shows the timing involved in the Interrupt Acknowledge signal, $\overline{INTA}$, which is a composite of two signals as shown below:

$$\overline{INTA} = \overline{IORQ} \times \overline{M1}$$

2. The $\overline{INT}$ line is sampled by the CPU during the negative-going trailing edge of the last clock pulse of the instruction. If it is active, i.e. low, it goes into an interrupt cycle as shown in Fig. 6-8.

3. At the leading edge of T1, $\overline{M1}$ goes low or active. During the trailing edge of the first Wait period clock pulse, TW1, $\overline{IORQ}$ goes low. Both these signals remain active or low for 1½ clock periods. They go high at the leading edge of the T3 clock pulse. Thus, during this 1½ clock period, $\overline{INTA}$ is active.

4. During $\overline{INTA}$ time, the PIO checks the highest priority port which is requesting the interrupt by determining which device has its IEI high and its IEO low.

5. The highest priority I/O device, that is requesting an interrupt service, places the word contained in its interrupt Vector Register on the CPU data bus, thereby identifying itself.

6. During the time when $\overline{M1}$ is active, no new interrupt requests are generated, thus giving the INTA composite signal ample time to ripple through up to 4 PIO daisy chain configurations.

7. Figure 6-9 shows how up to 4 chips can be connected to form the daisy chain.

6-5.6 Extending the Daisy Chain

1. Figure 6-9 shows that up to 4 PIO chips can be connected without any additional external logic.

2. This limitation is designed so that the IEO will have adequate time to ripple through the entire chain in the time available between the start of the $\overline{M1}$ signal and the leading edge of the $\overline{IORQ}$ signal during the $\overline{INTA}$ cycle. (See Fig. 6-8.)

3. In many applications it is desirable to extend the chain beyond the simple 4 shown

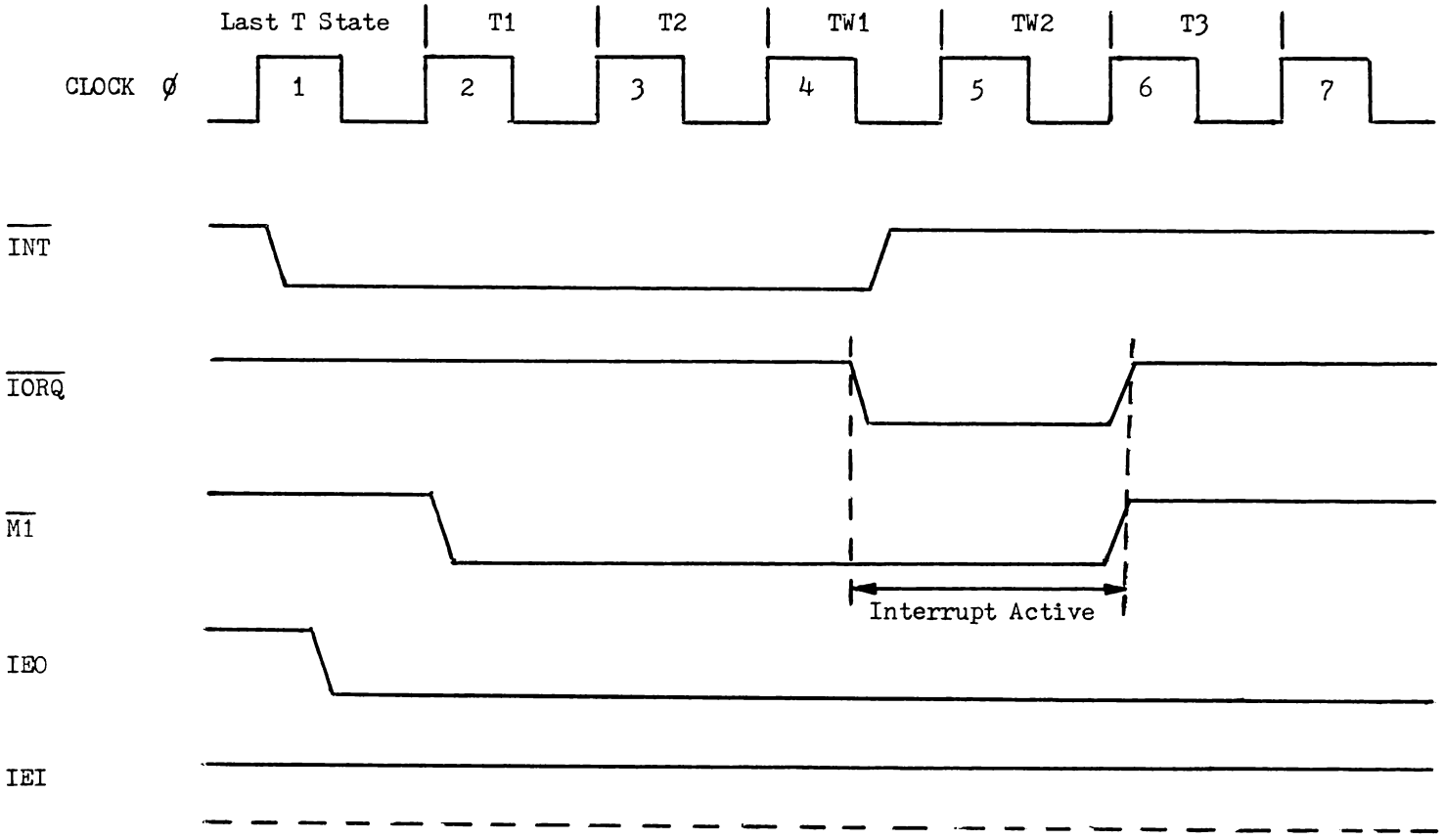


Fig. 6-8 Interrupt acknowledge timing.

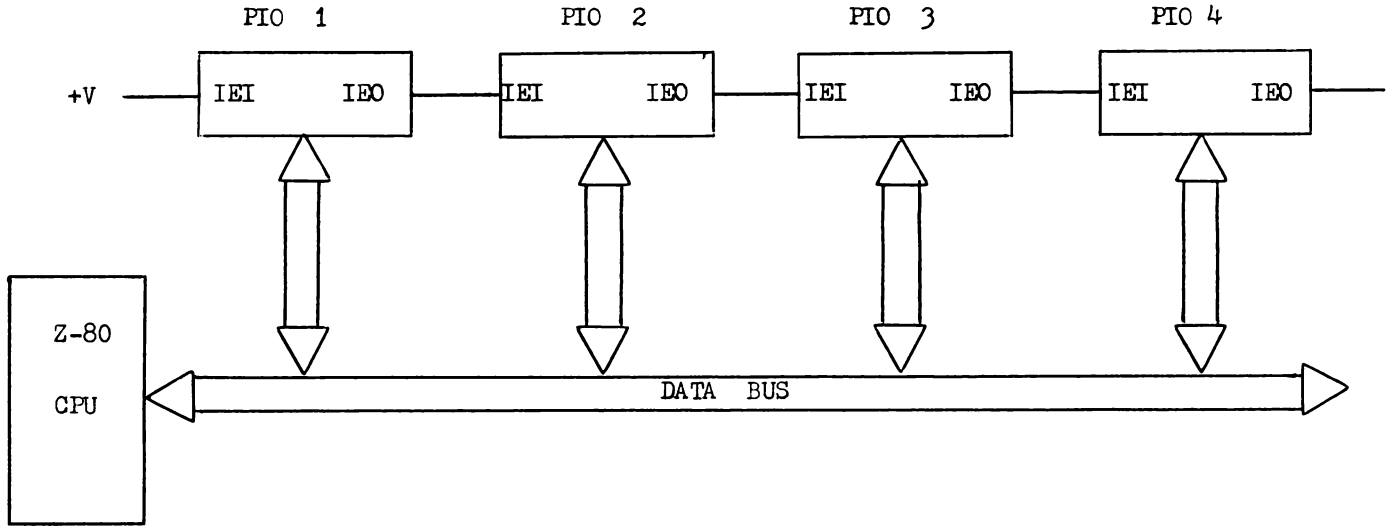


Fig. 6-9 Maximum PIO daisy chain without additional logic.

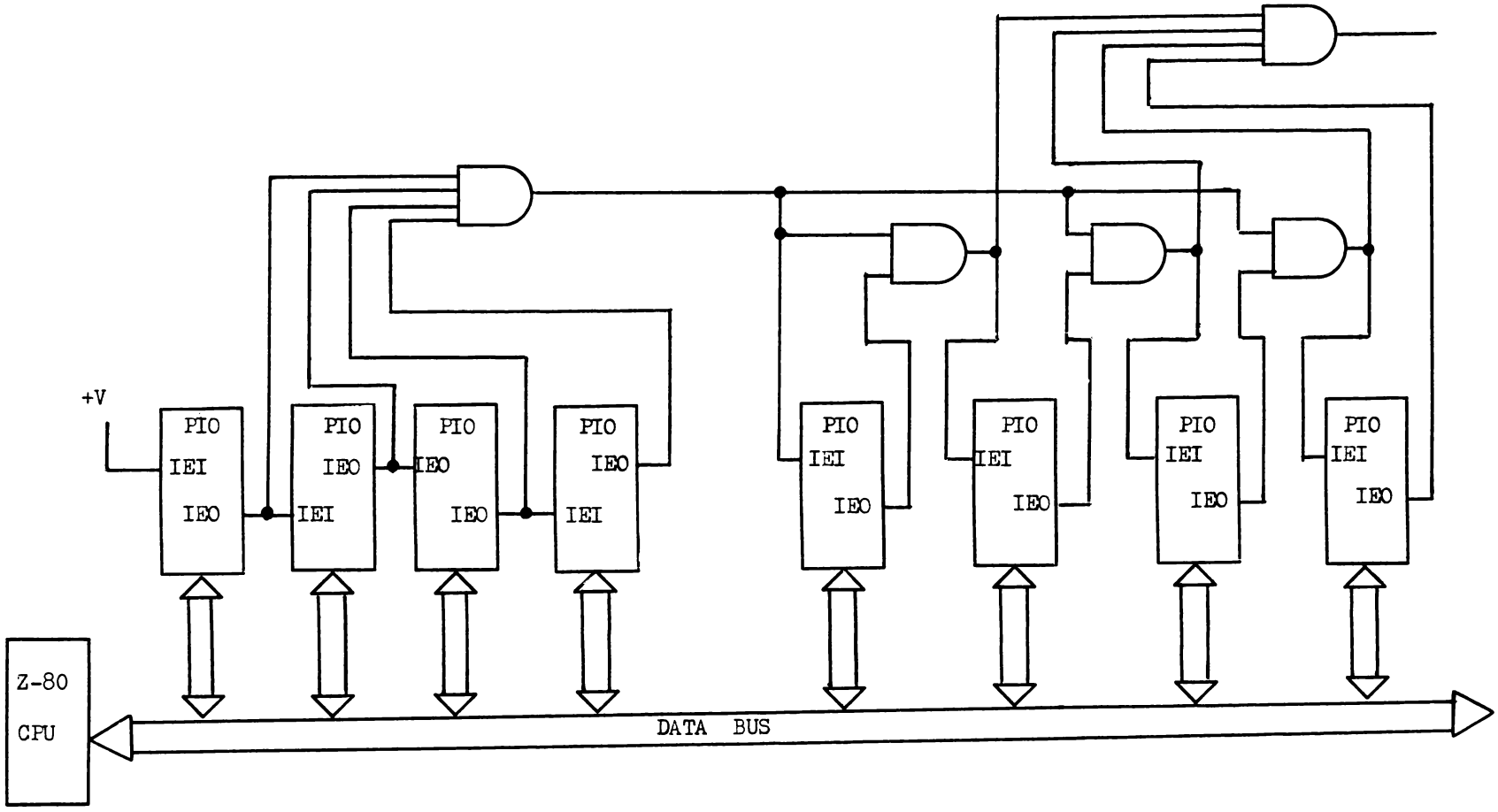


Fig. 6-10 Extending the interrupt priority daisy chain with standard TTL logic.

in Fig. 6-9. It is possible to extend this to 30 PIO chips or more, using standard TTL logic, by means of the look-ahead logic structure shown in Fig. 6-10, which requires 2-input and 4-input AND gates connected as shown.

6-6 REVIEW QUESTIONS

6-1 In the Z-80 system, data transfers between the CPU and the I/O devices are initiated by,

a. The CPU
b. The I/O devices
c. The PIO

6-2 Indicate true or false for the PIO.

a. ___ The PIO can be reconfigured under program control.

b. ___ The daisy chain technique is used to provide an interrupt priority system.

c. ___ Both Ports A and B have a bidirectional byte transfer mode (Mode 2).

d. ___ Both Ports A and B have handshake capabilities.

6-3 Refer to the Generalized Block Diagram of Fig. 6-1 and indicate true or false.

a. ___ Control information is transferred from the CPU to the PIO on an 8-bit Control Bus.

b. ___ Lines A0 - A7 are bidirectional byte lines and B0 - B7 are unidirectional Lines.

c. ___ The PIO communicates with the CPU via 3, unidirectional Interrupt Control Lines.

d. ___ The Internal Control Logic Block synchronizes the data flow between the CPU and the I/O devices.

6-4 Refer to the Port Logic block diagram of Fig. 6-2. The Operating mode of the PIO is, (circle all that apply).

a. Determined by the type of I/O used by that port.
b. Determined by the CPU.
c. Established by preloading a control word into the 8-bit I/O Select Register.
d. Established by means of a 2-bit code sent by the CPU.

6-5 Again refer to Fig. 6-2. The 8-bit I/O Select Logic is used in the

a. Bit Control Mode (Mode 3)

b. Output Byte Mode (Mode 0)
c. Bidirectional Byte Mode (Mode 2)
d. Input Byte Mode (Mode 1)

6-6 The following statements relate to the Mask Control Register of Fig. 6-2. Fill in the blanks.

a. It is preloaded with a 2-bit code from the _____. (CPU; I/O device)

b. It defines the desired active states for the _____ bits of the Mask Register. (Masked; Unmasked)

c. It defines that the interrupt should be generated by the AND function, that is, when _____ of the unmasked bits are in the _____ state. (All; some; Active; Inactive)

6-7 The following statements relate to the Interrupt Control Protocol in the PIO. Indicate true or false.

a. ___ Port A has a higher priority than Port B.

b. ___ The device closest to the CPU has the lowest priority.

c. ___ In the Bit Control Mode, an interrupt can be generated only when the I/O device matches a programmed value.

d. ___ An I/O device can never interrupt during the subroutine execution of any other device on the daisy chain.

6-8 The tristate, 8-bit, data bus _____ is active high and is used for data and commands transfers between the CPU and the PIO. (A0 - A7; B0 - B7; D0 - D7)

6-9 The $\overline{CE}$ signal is (circle all that apply.)

a. Active high and informs the PIO to transmit the data word to the CPU.
b. Active low and informs the CPU to transmit the command word to the PIO.
c. Active low and informs the PIO to transmit the data word to the CPU.
d. Active high and informs the CPU to transmit the data word to the PIO.

6-10 The following statements apply to the $\overline{M1}$ signal. Indicate true or false.

a. ___ It synchronizes the PIO interrupt cycle.

b. ___ It is a SYNC pulse from the PIO to the CPU.

c. ___ The CPU fetches an instruction from the memory when $\overline{M1}$ and $\overline{RD}$ are simultaneously active.

d. ___ The CPU acknowledges an interrupt when $\overline{M1}$ and $\overline{IORQ}$ are simultaneously active.

e. ___When $\overline{M1}$ is active by itself, the PIO goes into the Set state.

6-11 The following statements relate to the $\overline{A\ STB}$ signal when the PIO is operated in Mode 2, bidirectional Mode. Fill in the blanks.

a. Indicates that the _____ word from the output register of Port _____ is put on its own bidirectional data bus. (Command; data; A; B)

b. The _____ going, _____ edge of this pulse indicates that the data is received by the I/O device. (Positive; negative; leading; trailing)

6-12 Refer to the Mode 0, byte output, timing chart of Fig. 6-4. The $\overline{WR}$ is a composite of:

a. RD x $\overline{CE}$ x $\overline{C/D}$ x $\overline{IORQ}$

b. $\overline{RD}$ x $\overline{CE}$ x $\overline{C/D}$ x IORQ

c. RD x $\overline{CE}$ x C/D x IORQ

d. $\overline{RD}$ x CE x C/E x $\overline{IORQ}$

6-13 Again refer to Fig. 6-4. Indicate true or false for the READY signal.

a. ___ Goes low on the first trailing edge of the clock pulse after the trailing edge of $\overline{IORQ}$.

b. ___ This signal is low when the data in the port is changing.

c. ___ This signal will not go low until the negative-going edge of the clock pulse happens.

d. ___ Positive-going, trailing edge of $\overline{STROBE}$ signal indicates that the I/O has obtained the data.

6-14 The following statements relate to the Mode 1 timing chart of Fig. 6-5. Fill in the blanks.

a. The _____ cycle is initiated by the negative-going pulse on the _____ line. (Input; output; $\overline{INT}$; $\overline{STROBE}$)

b. The positive-going, trailing edge of the $\overline{STROBE}$ signal activates the _____ signal. (READY; $\overline{INT}$; $\overline{RD}$)

c. If the READY line is already active, it is forced into the low state 1½ clock periods following the leading edge of the _____ signal. ($\overline{INT}$; $\overline{IORQ}$; $\overline{STROBE}$; $\overline{RD}$)

6-15 Refer to the bidirectional byte timing chart of Fig. 6-6 and indicate true or false.

a. ___ The positive-going, trailing edge of the $\overline{A\ STB}$ signal latches the data word in the I/O device.

b. ___ The data into the PIO are latched into the input register when the $\overline{\text{B STB}}$ signal is high.

c. ___ The $\overline{\text{WR}}$ signal is a composite of RD x $\overline{\text{CE}}$ x $\overline{\text{C/D}}$ x $\overline{\text{IORQ}}$

d. ___ Mode 2 utilizes all 4 handshake lines and is available only for Port A of the PIO.

6-16 The following questions relate to the timing chart of the Bit Control Mode (Fig. 6-7). Answer yes or no.

a. ___ Are the 4 handshake signals used in this mode?

b. ___ Is the A RDY line forced into a high state?

c. ___ Is an interrupt generated if the words on the port data lines do not conform to the preloaded logical conditions?

6-17 The statements below refer to the Interrupt Acknowledge and servicing in the PIO. (See Fig. 6-8.) Indicate true or false.

a. ___ The $\overline{\text{INTA}}$ signal is composed of $\overline{\text{IORQ}}$ and $\overline{\text{M1}}$.

b. ___ The $\overline{\text{INT}}$ line is sampled by the CPU during the negative-going trailing edge of the last clock pulse of the instruction.

c. ___ During the trailing edge of the first WAIT period clock pulse, TW1, $\overline{\text{IORQ}}$ goes low.

d. ___ During the $\overline{\text{INTA}}$ time, PIO determines priority by checking for I/O device with high IEO and low IEI conditions.

e. ___ During the time when $\overline{\text{M1}}$ is active, no new interrupt requests are generated.

6-18 Extending the daisy chain requires the following additional gates; (circle the correct answer).

a. 4-input AND gates only.
b. 2-input AND and 4-input OR gates.
c. 2-input and 4-input OR gates.
d. 2-input and 4-input AND gates.
e. 2-input OR and 4-input AND gates.

7 PROGRAMMING THE PIO CHIP

CHAPTER OBJECTIVES

The objectives of this chapter are as follows:

1. To introduce the concept of customizing the PIO chip to specific applications.
2. To explain the format and functions of the Mode Set byte.
3. To explain the function and use of the I/O Select Register Control word for the Mode 3 operation.
4. To introduce the usage of the Interrupt Vector.
5. To discuss the function and format of the Interrupt Control byte.
6. To explain the function of the Mask Control byte.
7. To introduce the format of the Interrupt Enable Flip-flop Control byte.
8. To demonstrate how the PIO can be initialized for an industrial control application by means of a typical example.
9. To set up and program the PIO chip for a typical byte transfer application by means of a typical example.

TEXTBOOK REFERENCES

1. For general discussion of programmable parallel interfaces - Sec. 12-3
2. For operational modes control words - Secs. 12-3.3.2 and 12-3.3.3
3. For bit set/reset control word - Sec. 12-3.3.4
4. For discussion on Vectored Interrupt system - Sec. 9-5

7-1 INTRODUCTION

In Chapter 6 we had discussed the PIO chip and how it operates. This included the various modes of operation. The PIO is a programmable interface which means that under program control, the user can select and use the various features of the chip to customize the interface to the exact requirements of the particular application involved. This is of course true within certain specified limits of the design.

The process of customizing the PIO chip requires that the chip be first set up by means of an initialization program which can include up to six bytes. Once the PIO is set up by the initialization program, it remains in that condition until such time when it is altered by another similar initialization program. After power is turned off, the chip must be set up again with an initialization program before it can be used.

In this chapter we describe the initialization process and the bytes or words involved in such a program. Also, examples are included to show how the chip can be programmed for some typical applications.

7-2 THE PROGRAMMING WORDS

In this section we describe and discuss the formats of the various bytes (or words) involved in the initialization program. The programming bytes are presented in the sequence in which they would be included in the initialization program.

7-2.1 Word for Mode Set

As we discussed in Chapter 6, the PIO can be operated in any one of the four possible modes. The initialization program must identify which of the four modes is selected for a specific application. The first word in the set-up program does this. The format of this byte is shown in Fig. 7-1.

D7	D6	D5	D4	D3	D2	D1	D0
M1	M0	X	X	1	1	1	1

Fig. 7-1 Format of the Mode Set Byte.

In Fig. 7-1, the bits D0 - D3 are always 1. The bits in positions D4 and D5 are not relevant. They could be anything. An X in each position means don't care. The code for the selected mode is inserted in bit positions D7 and D6. The codes are given below.

M1	M0	Mode Selected
0	0	Output byte
0	1	Input byte
1	0	Bidirectional byte
1	1	Bit Control

When Mode 3 (i.e. the Bit Control Mode) is used, the next byte must be used to set up the I/O Select Register (See Fig. 6-2.) The format of this byte is shown in Fig. 7-2. The function of this word is to assign each line to the I/O Data/Control Bus individually as either an input or an output line. A 1 in any bit position sets the corresponding I/O line as an input line. A 0 sets the line as an output line.

D7	D6	D5	D4	D3	D2	D1	D0
I/O_7	I/O_6	I/O_5	I/O_4	I/O_3	I/O_2	I/O_1	I/O_0

Fig. 7-2 Format of the I/O Select Register Control word for the Mode 3 operation.

7-2.2 The Interrupt Vector

Since the Z-80 system uses the vectored interrupt scheme for identifying the interrupting device, in conjunction with the daisy chain for prioritization, it is necessary that a vector or a unique identifying code (i.e. an address) be assigned to each I/O chip. A 7-bit vector can be assigned as shown in Fig. 7-3. Notice that the D0 bit in this byte contains a 0. The selection of the code or the bit combination is left entirely to the discretion of the user but, it must be loaded only in bit positions D1 - D7.

D7	D6	D5	D4	D3	D2	D1	D0
V7	V6	V5	V4	V3	V2	V1	0

Fig. 7-3 Format of the Interrupt Vector.

7-2.3 Set Interrupt Control Byte

The next byte in the program sets up the control word for the Interrupt Control Logic block shown in Fig. 6-1. The format of this word is shown in Fig. 7-4. In this word, bits D3 - D0 are always 0111.

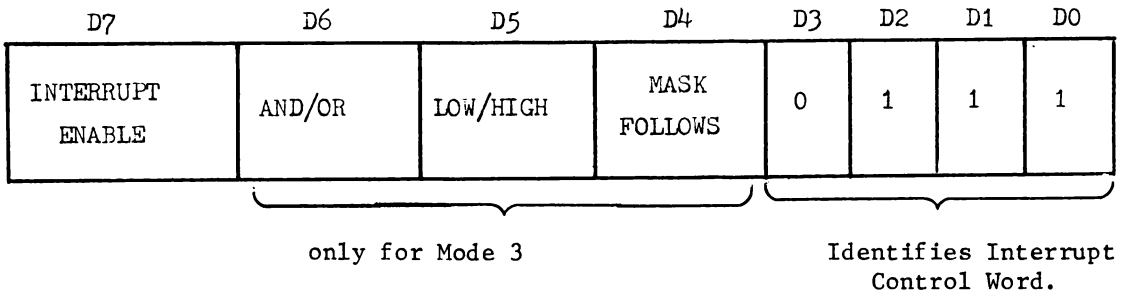


Fig. 7-4 Format of the Interrupt Control Byte.

A 1 in the D7 position sets the interrupt enable flip-flop of the selected port. This means that the particular port could generate an interrupt. A 0 in this position resets this flip-flop and so interrupts cannot be generated by this port.

Bits D6, D5 and D4 are used only if the PIO is initialized for the Mode 3 operation. Bit D6 enables the programmer to define the specific conditions that must exist on the I/O lines for that port to generate an interrupt. A 1 in the D6 position indicates an AND function which means that all I/O lines must go to the specified logical states for an interrupt to be generated. A 0 in D6 specifies an OR condition which means that an interrupt will be generated if any one or more of the I/O lines go to the specified logical state.

Bit position D5 specifies the high or low state of the I/O which will be monitored for generating interrupts. A 1 in D5 will monitor the I/O lines of that port for the high logical state and a 0 will monitor them for the low state.

At the option of the programmer, it is possible to monitor only certain desired I/O lines and mask out the rest. Bit position D4 allows us to do this. A 1 in this position indicates that the next control byte, inputted into the PIO, is the Mask Word whose format is shown in Fig. 7-5. A 0 in any bit position of the Mask Word will monitor the corresponding I/O line while a 1 will mask out the corresponding line.

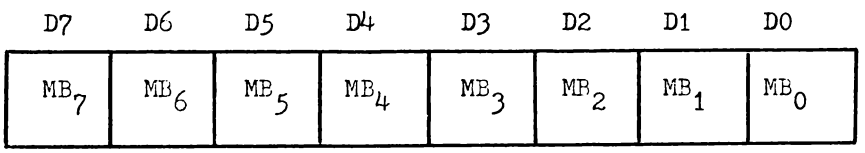


Fig. 7-5 Format of the Mask Control byte.

7-2.4 Interrupt Enable Flip-Flop Set

The interrupt enable flip-flop can be set or reset by means of the byte whose format is shown in Fig. 7-6. In this case a 1 sets the flip-flop and a 0 resets it. Bit positions D3 - D0 are always 0011 and this code identifies the byte as the interrupt enable control word. Bits D6, D5 and D4 can be any combination of is and 0s.

D7	D6	D5	D4	D3	D2	D1	D0
INT. ENABLE	X	X	X	0	0	1	1

Fig. 7-6 Format of the Interrupt Enable Flip-flop control byte.

We will now show some examples of how the PIO can be initialized for specific applications.

7-3 PROGRAMMING THE PIO FOR A CONTROL APPLICATION

Example 7-1

In an industrial control application, the RPM of an electric motor must be maintained within the upper RPM limit, R_{max}, and the lower limit, R_{min}. In other words,

$$R_{max} > RPM > R_{min}$$

An RPM value lower than R_{min} is indicated by a digital 1 on signal line S_1 and RPM value higher than R_{max} is indicated by a 1 on the signal line S_2. If the motor stalls, because of a sudden surge in the load, the stalled condition is indicated by a digital 1 on the S_3 signal line. Finally, an equipment failure in the motor controller is indicated by a 1 on the S_4 line. From the computer side, the following signals are to be sent to the motor controller. A high state is used for generating interrupts.

1. Special test signal, ST, to initiate the test cycle.
2. Power turn on signal, P_{ON}.
3. Increase motor RPM signal, R_I.
4. Decrease motor RPM signal, R_D.

The unique address assigned to the PIO chip is 11110100. Draw a system configuration for this application and write a PIO initialization program, explaining how each program byte is constructed. Write the program in both machine language and hexadecimal codes.

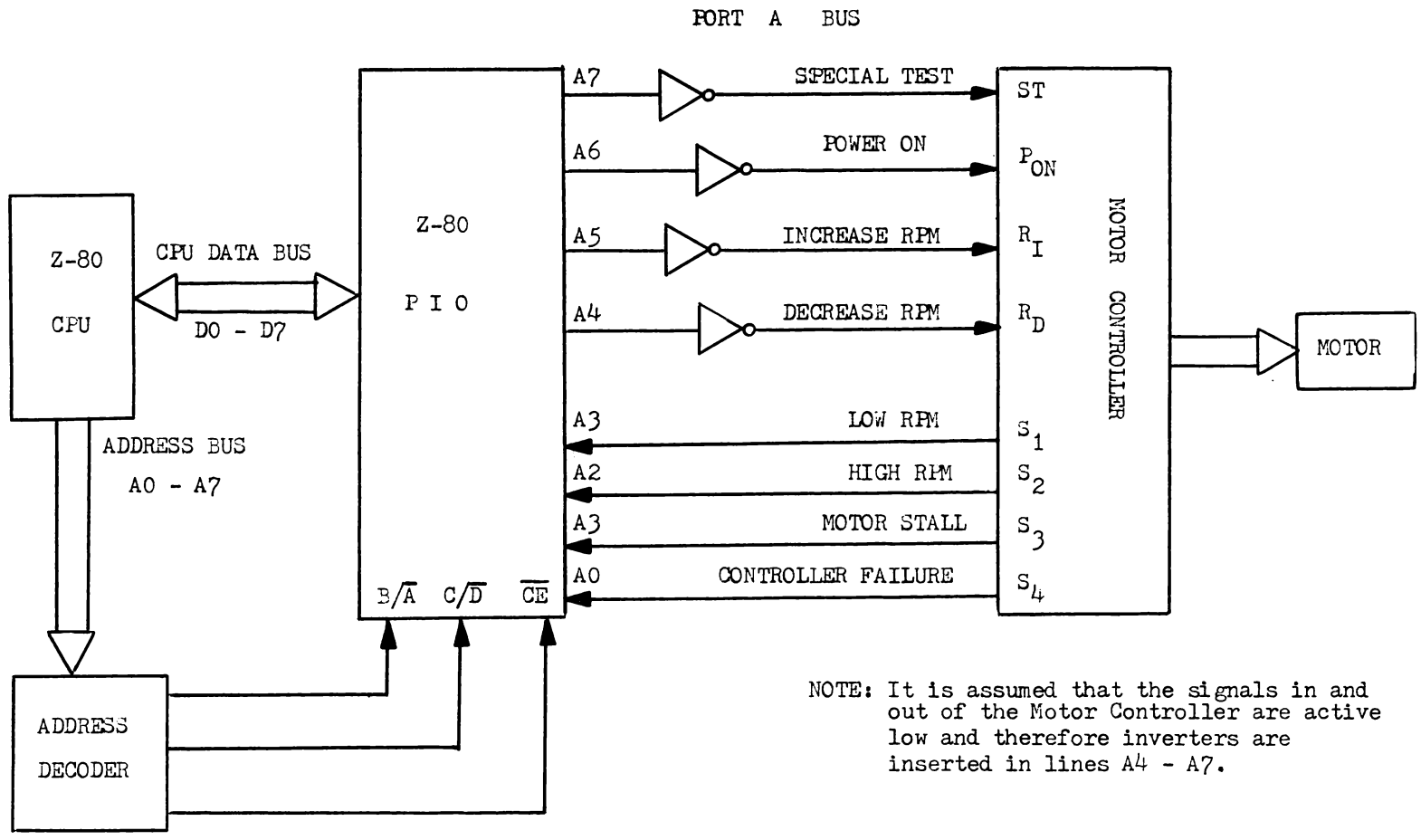


Fig. 7-7 System configuration for Example 7-1 (Mode 3 - bit control mode)

Solution

From the problem statement it is clear that we will have to use Mode 3 (Bit Control Mode) in this particular application. The system configuration for this problem is shown in Fig. 7-7.

For this particular application the status and control word, be transferred between the CPU and the PIO on the D0 - D7 bus, is formatted as shown in Fig. 7-8. Note that the status and control bits are arbitrarily assigned the various bit positions in this word.

D7	D6	D5	D4	D3	D2	D1	D0
SPECIAL TEST (ST)	POWER ON (P_{ON})	INCREASE RPM (R_I)	DECREASE RPM (R_D)	LOW RPM (S_1)	HIGH RPM (S_2)	MOTOR STALL (S_3)	CONTROLLER FAILURE (S_4)

Fig. 7-8 Format of the status and control word for example 7-1.

The following procedure is then used for generating the initialization program.

1. We use Port A of the PIO for the Mode 3 operation. Mode 3 is selected by loading the Mode Select word from the CPU into the PIO, as discussed in Sec. 7-2.1. The format of this word is shown in Fig. 7-9.

D7	D6	D5	D4	D3	D2	D1	D0
1	1	X	X	1	1	1	1

Fig. 7-9 Mode Set byte for example 7-1.

2. Because Mode 3 is selected, the next word in the program, to be sent by the CPU is the I/O Select word which was discussed in Sec. 7-2.1, Fig. 7-2. This word assigns the various PIO-to-Motor Controller lines (A0 - A7) as either input or output lines. In this case, the direction of signal flow is defined in Fig. 7-7, so the word shown in Fig. 7-10 is loaded into the I/O Select Register of Fig. 6-2.

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	1	1	1	1

Fig. 7-10 I/O Select Word for example 7-1.

3. The Interrupt Vector of the PIO is now loaded. The format of this byte, which was previously discussed in Sec. 7-2.2, is shown in Fig. 7-11 for this problem.

D7	D6	D5	D4	D3	D2	D1	D0
1	1	1	0	1	0	1	0

Fig. 7-11 The Interrupt Vector for Example 7-1.

4. The Interrupt Control Byte, whose format was discussed in Sec. 7-2.3 (Fig. 7-4), is now constructed as shown in Fig. 7-12 where,

   a. Bit D7 is a 1 because the interrupt enable flip-flop is to be set.
   b. D6 is a 0 because any one of the incoming signals, S_1 - S_4, must be able to generate an interrupt. An OR function is used.
   c. D5 is a 1 because the incoming signals S_1 - S_4 are active high according to the problem statement.
   d. D4 is a 1 because the next byte in the initialization program is a mask byte.

D7	D6	D5	D4	D3	D2	D1	D0
1	0	1	1	0	1	1	1

Fig. 7-12 Set Interrupt Control byte for example 7-1.

5. The next word to be loaded is the Mask Word which was discussed in Sec. 7-2.3, Fig. 7-5. Since lines A3, A2, A1 and A0 are to be monitored, the format of the word is shown in Fig. 7-13. Note that bits D0 - D3 are zeros so lines A0 - A3 will be monitored and the rest masked out.

D7	D6	D5	D4	D3	D2	D1	D0
1	1	1	1	0	0	0	0

Fig. 7-13 The Mask Word for example 7-1.

6. This completes the words required to configure the PIO chip for our specific problem. Now if one of the sensors from the motor controller puts a 1 signal on any of the four lines A0 - A3, an interrupt will be generated and a status word will be sent by the PIO to the CPU. The CPU will then send back an appropriate control word to the motor controller via the PIO chip. The program for initializing the PIO for this problem is shown in Table 7-1. Both machine codes and hexadecimal codes are included.

Table 7-1 Initialization program for problem 7-1.

BYTE FUNCTION	HEXA-DECIMAL CODE	MACHINE LANGUAGE CODE							
		D7	D6	D5	D4	D3	D2	D1	D0
Mode Set Byte	C F	1	1	0	0	1	1	1	1
I/O Select Word	0 F	0	0	0	0	1	1	1	1
Interrupt Vector	E A	1	1	1	0	1	0	1	0
Set Interrupt Control	B 7	1	0	1	1	0	1	1	1
Mask Word	F 0	1	1	1	1	0	0	0	0

7-4 PROGRAMMING THE PIO FOR A BYTE TRANSFER APPLICATION

In this section we will show, by means of an example, how the PIO can be programmed for an application which requires the transfer of data bytes between a bidirectional I/O peripheral and the PIO. This means that the PIO must be operated in Mode 2, bidirectional mode. Recall from Chapter 6 that only Port A can be used in Mode 2 because

bidirectional data transfers utilize all four handshake lines. The PIO chip is designed such that in Mode 2 operation, the Port A handshake signals, i.e. A RDY and A STB, are used for output control and the Port B handshake signals, i.e. B RDY and B STB, are used for input control. Example 7-2 shows how the PIO can be programmed for this mode and how the system is configured for such an application.

Example 7-2

A bidirectional, 8-bit parallel I/O peripheral whose Interrupt Vector is 9C in hexadecimal code, is connected to Port A of the PIO chip. Draw a sketch of the system configuration. Write the program for initializing the PIO chip and explain how it is constructed. Show both machine language and hexadecimal codes for this problem.

Solution

The system diagram for this problem is shown in Fig. 7-14. Notice that the Port A handshake signals are used for output control signals and Port B handshake signals for the input control signals.

In Fig. 7-14, two inverters are inserted in the A RDY and the B RDY lines. This is because it is assumed that all handshake signals in and out of the I/O peripheral are active low.

To initialize the PIO chip, the Mode Set Word (Fig. 7-1) is loaded first. This is shown in Fig. 7-15.

D7	D6	D5	D4	D3	D2	D1	D0
1	0	0	0	1	1	1	1

Fig. 7-15 Mode Set word for example 7-2.

The Interrupt Vect is the next word to be loaded into the PIO. The problem statement gives this word to be 9C and so this word is shown in Fig. 7-16.

D7	D6	D5	D4	D3	D2	D1	D0
1	0	0	1	1	1	0	0

Fig. 7-16 Interrupt Vector word for example 7-2

Finally, the Interrupt Enable Flip-flop control byte, whose format is shown in Fig. 7-6, must be loaded. This is shown in Fig. 7-17.

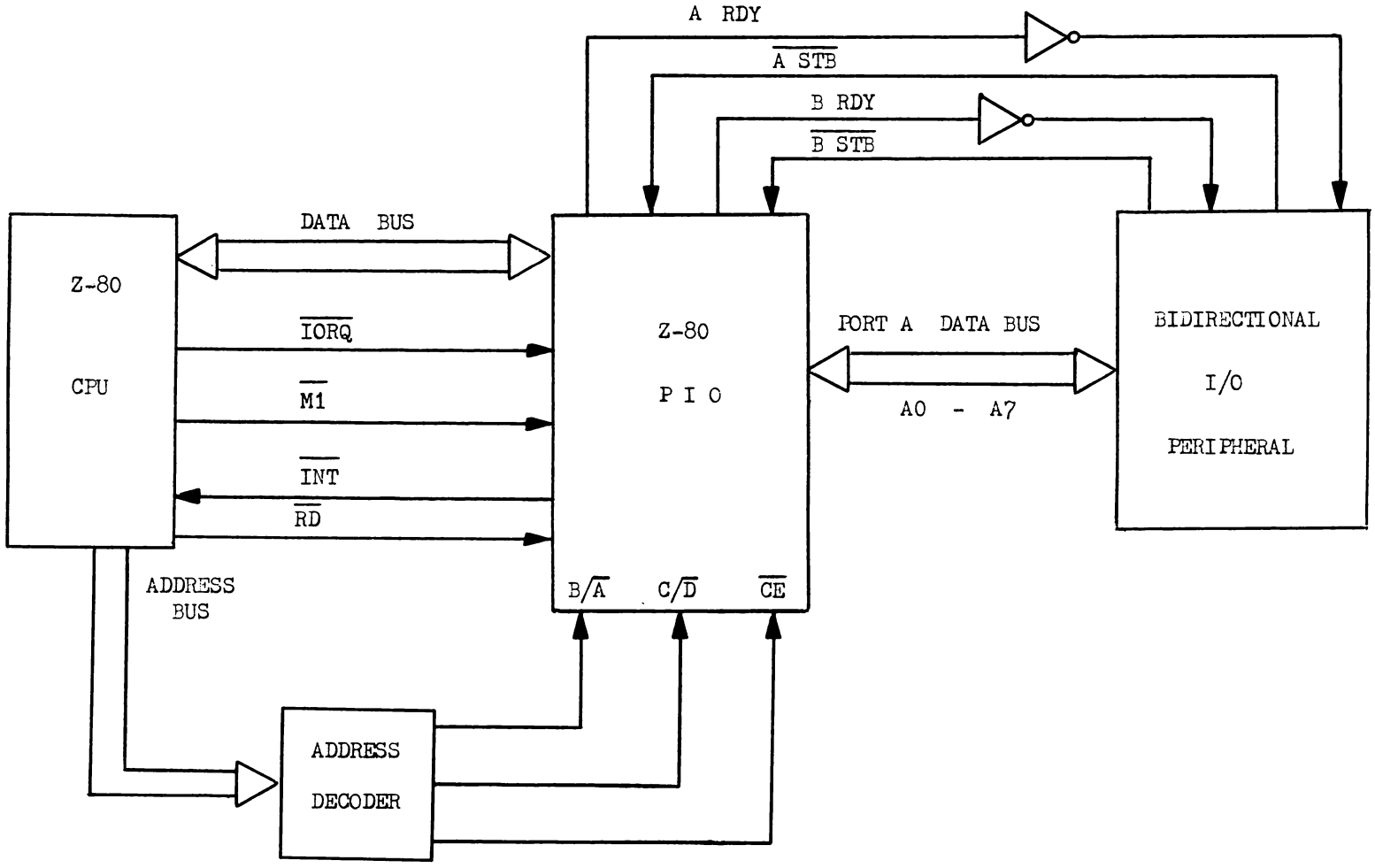


Fig. 7-14 System configuration for example 7-2.

D7	D6	D5	D4	D3	D2	D1	D0
1	0	0	0	0	0	1	1

Fig. 7-17 Interrupt Enable Flip-flop control byte for example 7-2.

The program for initializing the PIO for this problem is shown in convenient tabular form in Table 7-2. Both machine language codes and hexadecimal codes are included.

Table 7-2 Initialization program for example 7-2.

BYTE FUNCTION	HEXA-DECIMAL CODE	MACHINE LANGUAGE CODE							
		D7	D6	D5	D4	D3	D2	D1	D0
Mode Set Byte	8 F	1	0	0	0	1	1	1	1
Interrupt Vector	9 C	1	0	0	1	1	1	0	0
Interrupt Enable Flip-flop control byte	8 3	1	0	0	0	0	0	1	1

7-5 REVIEW QUESTIONS

7-1 The following statements relate to the programmable features of the PIO chip. Indicate true or false.

a. ___ The PIO can be customized to the specific requirements of a particular application.

b. ___ The initialization program always contains 6 bytes.

c. ___ After being initialized, the PIO remains in that configuration until re-initialized by a different program.

7-2 From the statements given below pick out the one(s) applicable to the initialization program byte 1 1 X X 1 1 1 1.

a. This is the I/O Select Register Control byte.
b. This is the Mode Set byte.
c. This byte defines the bidirectional byte transfer mode.
d. This byte must be followed by the I/O Select Register Control word.

7-3 In a PIO initialization program using Mode 3 operation, the second byte in the program is 1 0 1 0 1 0 1 0. Indicate true or false for the following statements which apply to this byte.

a. ___ It is the Interrupt Vector.

b. ___ Lines D0, D2 and D4 are input lines.

c. ___ Lines D1 and D7 are input lines.

d. ___ Line D6 is an output line and D3 is an input line.

7-4 From the following Interrupt Vectors (given in hexadecimal codes) pick out those that can be legitimately used in initializing the PIO chip.

a. 8 8
b. D 0
c. 9 B
d. E E
e. F F
f. A 6

7-5 The Interrupt Control byte of a particular PIO initialization program is 1 0 0 0 0 1 1 1 for Mode 3 operation. Indicate true or false for the following statements which relate to this byte.

a. ___ A low state of the I/O lines will be monitored for generating interrupts.

b. ___ The interrupt enable flip-flop is reset.

c. ___ The next word inputted into the PIO is Mask word.

d. ___ An interrupt will be generated if any one or more of the I/O lines go to a low state.

7-6 Construct the Interrupt Control byte for the initialization program that will satisfy the following conditions:

a. Monitor the high state of the I/O lines.
b. Set the interrupt enable flip-flop.
c. Define only specific I/O lines to be monitored.
d. Monitor the high state of all the specified I/O lines.

Bit positions	D7	D6	D5	D4	D3	D2	D1	D0
Byte Code								

7-7 Refer to question 7-6, Construct the byte which will follow the byte resulting from question 7-6 and which will monitor I/O lines D0, D1, D3, D5 and D6.

Bit positions	D7	D6	D5	D4	D3	D2	D1	D0
Byte code								

7-8 Refer to example 7-1 (Fig. 7-7). Inverters are inserted in PIO lines A4 - A7 because,

a. Signals A4 - A7 are outputted by the PIO chip.
b. Signals A3 - A0 do not have inverters external to the PIO chip.
c. Signals A4 - A7 are unidirectional signals.
d. Signals in and out of the motor controller are active low.
e. Signals in and out of the motor controller are active high.

7-9 Refer to the status and control word format for example 7-1 and indicate true or false.

a. ___ A single bit for indicating motor RPM (such as 1 for high and 0 for low RPM) could have been used instead of two signals.

b. ___ Signal S_4 from the motor controller is not really necessary because the controller would fail only if there were a power failure.

7-10 Refer to Fig. 7-8 which is the status and control byte for example 7-1. From the following statements pick out those that apply to this byte.

a. The status signals must always be placed on the lower half of the bus for Port A, i.e. on lines A0 - A3.
b. The control signals must always be outputted on the upper half of the Port A bus, i.e. on lines A4 - A7.
c. The assignment of the status and control lines of Port A is arbitrary.
d. No more than 4 lines may be used for incoming status signals.
e. No more than 4 lines may be used for the outgoing control signals.

7-11 Bit D6 of the Interrupt Control byte for example 7-1 is a 0 because,

a. The low state of the I/O lines are monitored.
b. Any incoming status signal is required to generate an interrupt.
c. The high state of the I/O lines are monitored.
d. A Mask byte follows this byte.

7-12 Indicate true or false for the Mask byte for example 7-1.

a. ___ This word is included in the initialization program because bit D4 of the previous Interrupt Control byte is a 1.

b. ___ If bit D4 of the previous Interrupt Control byte had been a 0, then the Mask Word of Fig. 7-13 will contain all 0s.

c. ___ Bits D4 - D7 are all is because lines A4 - A7 of the PIO Port A are output lines.

7-13 The following statements relate to the system configuration (Fig. 7-14) for example 7-2. Indicate true or false.

a. ___ Lines A7 - A4 are used for control signals.

b. ___ Lines A3 - A0 are used for status signals.

c. ___ Signals A RDY and $\overline{\text{A STB}}$ are used for input control.

d. ___ Signals B RDY and $\overline{\text{B STB}}$ are used for output control.

7-14 Refer to Fig. 7-14. Inverters are inserted in the A RDY and the B RDY lines because,

a. They are input control lines.
b. They are output control lines.
c. They indicate the status of the Z-80 CPU.
d. All handshake signals in and out of the peripheral are assumed to be active low.

7-15 Indicate true or false for example 7-2.

a. ___ The PIO could have used Mode 3 for the operation.

b. ___ Either Mode 0 or Mode 2 could have been selected.

c. ___ Only Mode 2 could have used in this application.

d. ___ Either Mode 1 or Mode 2 could have been selected.

7-16 The initialization program for example 7-2 contains only 3 bytes because,

a. Mode 2 requires only the Mode Set byte, the Interrupt Vector and the Interrupt Enable Flip-flop control byte.
b. Only one byte of data is transferred at any one time.
c. Only 4 handshake signals are required for example 7-2.
d. Bits D4 and D5 of the Mode Set byte are both 0s.

8 Z-80 COUNTER/TIMER CIRCUIT CHIP

CHAPTER OBJECTIVES

1. To introduce students to programmable timing and counting functions.
2. To present the 4 independent, programmable, channels for timing and counting in the Z-80 μC system.
3. To describe the operation of the counter mode and the decrement logic triggering by an external signal.
4. To discuss the operation of the timer mode as well as the prescaler and the down counter.
5. To show how the counter/timer circuit chip fits into and operates in the prioritized interrupt daisy chain of the Z-80 μC system.
6. To present the interrupt control logic and the protocol involved in the counter/timer operation.

TEXTBOOK REFERENCES

The subject of counter and/or timer circuits is not covered in the main hardcover textbook because it was felt that these subjects are very adequately covered in many other excellent textbooks on digital logic. Students are encouraged to refer to them. One book that this author has found particularly useful is "Practical Digital Design Using IC's" by Joseph D. Greenfield, A John Wiley publication.

8-1 INTRODUCTION

The Counter/Timer Circuit chip (CTC) provides 4 independent, programmable channels in the Z-80 based μC system for counting and timing functions. Under program control, the CTC can be reconfigured to operate in several different conditions and modes so that it can be interfaced with a wide variety of I/O devices.

When the CTC is used in the counter mode, an external signal is used to trigger the decrement logic. It is possible to have the CTC generate an interrupt when a certain number of external events have been counted out. In the timer mode, the CTC generates time intervals by dividing the system clock by means of a prescaler that decrements a preset down-counter. Time intervals as short as 6.4 μs (for the Z-80 system) or 4.0 μs (for the Z-80A system) can be generated. The time interval is an integer multiple of the clock period.

The four CTC channels have priorities established so that they can readily fit into the standard Z-80 interrupt daisy chain, previously described in Chapter 6. Channel 0 has the highest priority and Channel 3 the lowest.

8-2 PRINCIPAL FEATURES OF THE CTC

1. The CTC is a third generation chip which uses the N-channel, depletion mode, silicon-gate technology.

2. A single, +5 ±5% volts power supply is required.

3. A single-phase, 5-volt clock is used.

4. The chip operates in a temperature environment of 0°C to +70°C.

5. All inputs and outputs are fully TTL compatible.

6. The CTC can be interfaced directly with the Z-80/CPU.

7. The CTC can be directly interfaced with the Z-80/SIO chip. (to be described in Chapter 10) for generating the desired baud rate.

8. When used in either the counter or the timer mode, a down counter is provided to indicate to the CPU the number of counts still to go before zero.

9. A Time Constant Register makes it possible to reload the Down Counter when it reaches zero. This can be done in either the counter or the timer mode.

10. Either a positive or a negative trigger can be selected to initiate timer operation in the timer mode or to monitor for the events count in the counter mode.

8-3 THE CTC ARCHITECTURE

8-3.1 Generalized CTC Block Diagram

1. Figure 8-1 shows a generalized block diagram of the CTC.

2. On the computer side, the CTC chip provides I/O interface logic for transferring

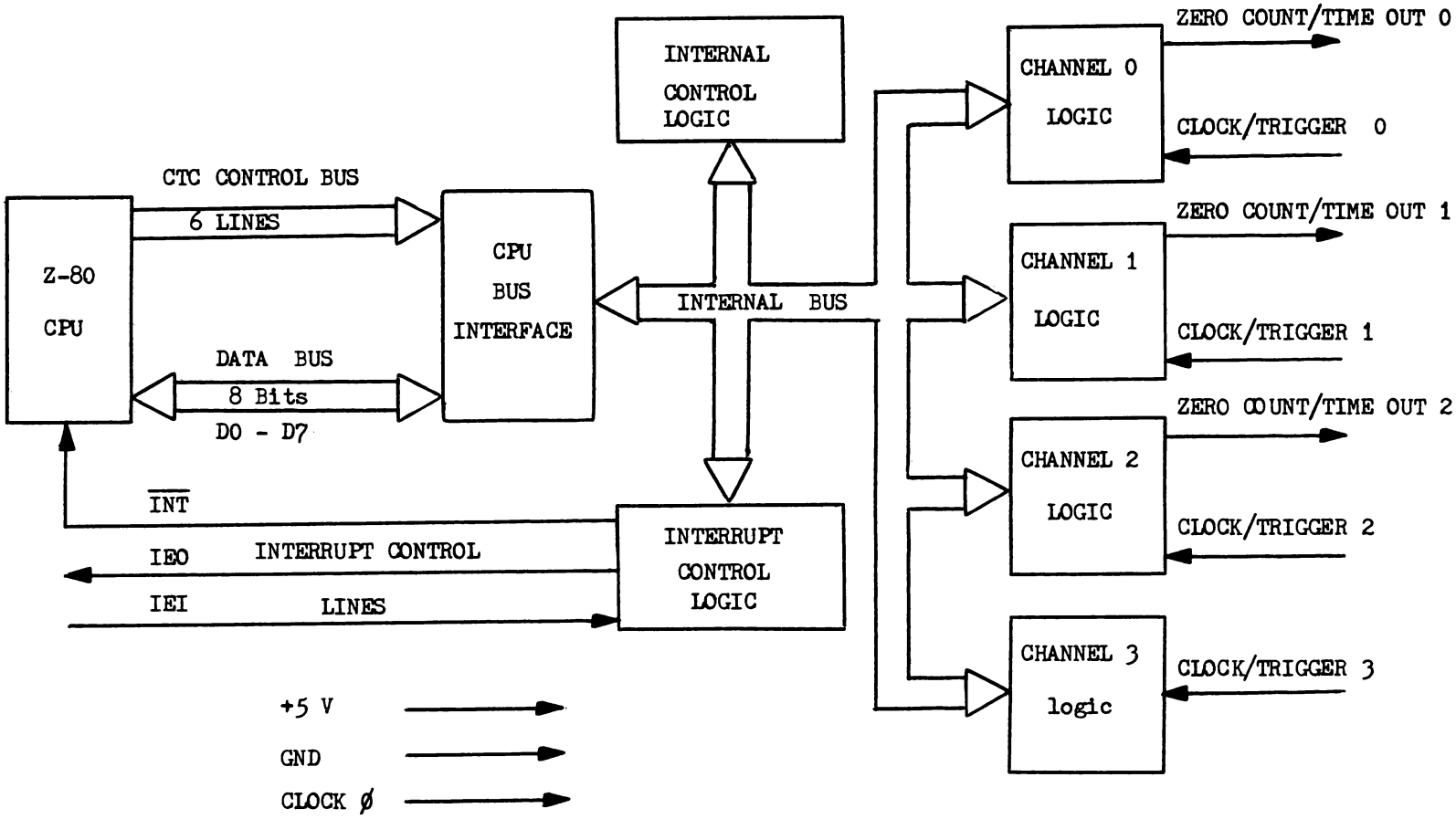


Fig. 8-1 Generalized CTC block diagram.

data on an 8-bit bus, D0 - D7. A 6-bit control bus is also present.

3. Four independent channels are numbered 0 through 3. Channel 0 has the highest priority and channel 3 the lowest. These four channels can be connected into the 4 consecutive slots of the standard Z-80 daisy chain which was previously discussed in Chapter 6.

4. A unique interrupt vector can be generated for each channel. This vector is used to determine the starting address for that particular channel's service subroutine.

5. Also on the computer side, the CTC communicates via three interrupt vector control lines. Each of these lines is an unidirectional line.

6. The Internal Control Logic block primarily deals with the overall operation of the CTC chip and includes functions such as the read/write operation, chip enable and reset functions.

7. The Interrupt Control Logic works in a manner very similar to that described in Chapter 6 for this block. Fundamentally, it performs the following functions:

   a. If the IEI is high, the CTC has the priority.
   b. During execution of the interrupt service subroutine, the IEO signal is held low. This inhibits interrupt requests from I/O devices with lower priorities.

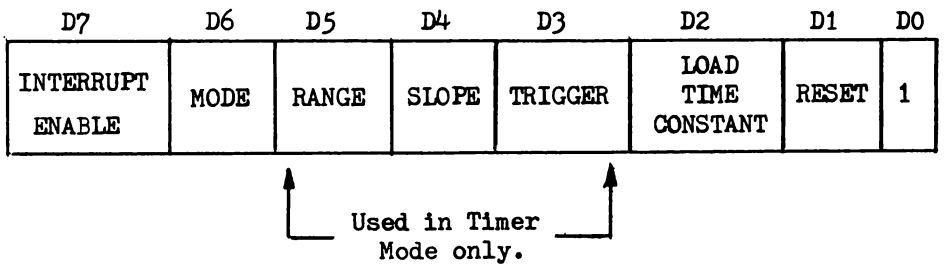
8-3.2 The Channel Logic Block Diagram

1. Figure 8-2 is a block diagram of the block labelled Channel Logic in Fig. 8-1. It consists of 2 counters, 2 registers and the associated control logic.

2. The Channel Control Register and Logic receives an 8-bit control word from the CPU when that particular channel is programmed or selected.

3. The selection of one of the 4 channels is made by a 2-bit code supplied by the CPU and inputted to the CTC chip on two pins (CS0 and CS1) which are connected to the A0 and A1 lines of the address bus from the CPU. The channel codes are shown below in table form.

4. The format of the control word sent by the CPU, and loaded into the Channel Control Register, is shown below:



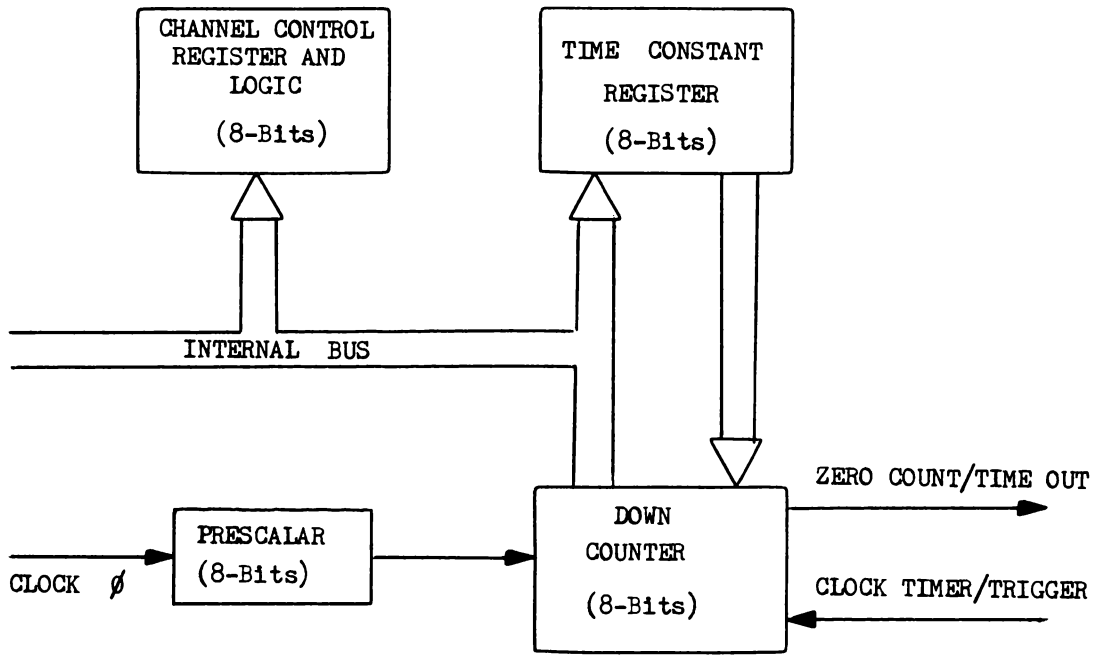


Fig. 8-2 Channel Logic block diagram.

Table 8-1 Channel select codes.

	CS1	CS0
Channel 0	0	0
Channel 1	0	1
Channel 2	1	0
Channel 3	1	1

The bits of the channel control word have the following meanings:

D0 - 1 indicates control word
0 indicates vector
D1 - 1 software reset
0 continued operation

D2 - 1 time constant follows
0 no time constant follows
D3 - 1 clock/trigger pulse starts the timer
0 automatic trigger when time constant is loaded
D4 - 1 selects positive-going edge of clock/trigger pulse
0 selects negative-going edge of clock/trigger pulse
D5 - 1 indicates prescaled value of 256
0 indicates prescaled value of 16
D6 - 1 indicates counter mode
0 indicates timer mode
D7 - 1 enables interrupt
0 disables interrupt

5. The Prescaler, which is used only in the Timer Mode, is an 8-bit register which is programmed by the CPU (through the Channel Control Register).

6. The Prescaler divides the inputted clock frequency (i.e. the system clock Ø) by either 16 or 256. The divided output is used as the clock input to the down counter.

7. The Time Constant Register is an 8-bit register which is used both in the counter and the timer operation modes. As noted in paragraph 4 above, a 1 in D2 in the channel control word indicates that the next word sent to the selected channel by the CPU is a time constant word, and it is preloaded in the Time Constant Register. This word is an integer time constant whose value ranges from 0 to 255 since it is 8 bits long. The 0 is interpreted as 256.

8. When the CTC chip is first initialized, this time constant value is loaded into the down counter. After the down counter is decremented to zero, the value in the Time Constant Register is automatically reloaded into the down counter.

9. If a new integer is loaded into the Time Constant Register while that particular channel's down counter is in the process of counting down, the count down is not interrupted. The new time constant is loaded into the down counter only after it reaches zero.

10. The down counter is an 8-bit register/counter which is always decremented when the CTC is used in either the Counter or the Timer Mode. It receives its clock Ø from the divided value outputted by the 8-bit Prescaler. A high signal is outputted on the Zero Count/Time Out line every time the count reaches zero in this counter.

11. Bit D3 of the Channel Control Word determines how the down counter is triggered to start counting, i.e. decrementing. If D3 is a 0, the counter is triggered automatically and starts decrementing. Upon reaching a count of zero, the down counter is reloaded with the starting value (i.e. the time constant) from the Time Constant Register. Once started, this process continues until it is stopped by a reset.

12. If Bit D3 of the Channel Control Word is a 1, the decrementing process of the down

counter is started only when the active edge of the pulse appears on the Clock Timer/Trigger line as shown in Fig. 8-2. Again, once it is started, it will continue running until it is stopped by a reset.

13. In the generalized block diagram of the CTC chip, note that the Zero Count/Time Out signal is available from channels 0, 1 and 2 but not from channel 3. This is due to the package pin limitations. Thus, channel 3 can only be used in those applications which do not require this signal output from the CTC.

14. An unusual feature of the CTC is that the CPU can obtain, or read, the contents of the down counter any time by performing an I/O operation of the selected CTC channel. This feature enables the CPU to determine how many count downs are still to go before the down counter reaches zero.

8-3.3 Interrupt Control Logic and Protocol

1. The primary function of the Interrupt Control Logic in the CTC chip (see Fig. 8-1) is to ensure that the interrupt protocol of the Z-80 system is followed by the CTC.

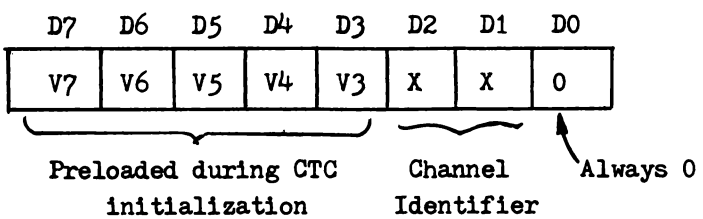
2. Again, because of the daisy chain technique, the priority of any device depends upon its relative position in this chain. The device closest to the CPU has the highest priority.

3. Within the CTC chip itself, Channel 0 has the highest priority and Channel 3 the lowest. Note that in accordance with the Z-80 interrupt protocol, higher priority channels can override and interrupt lower priority channels but not vice versa.

4. Any channel can be programmed to issue an interrupt request when its down counter reaches a zero count. However, this requires that the CPU be programmed for the interrupt Mode 2.

5. Upon receiving the interrupt acknowledge from the CPU, the CTC logic determines which of the highest priority channels is interrupting and will be serviced. If the IEI signal is active, thereby indicating that the CTC has priority within the daisy chain, the CTC will place its 8-bit interrupt vector on the data bus.

6. The format of the CTC interrupt vector is shown below. Note that this word is identified by bit D0 being always zero as opposed to the previously discussed channel control word whose D0 bit is always a 1.



7. Bits D3 - D7 of the interrupt vector are preloaded during the initialization of the CTC chip. Bits D2 and D1 represent a 2-bit code identifying the interrupting channel. These two bits are generated and inserted into the vector by the Interrupt Control Logic of the CTC. The table below shows the codes for bits D2 and D1.

Table 8-2 Interrupting channel identification codes.

	D2	D1
Channel 0	0	0
Channel 1	0	1
Channel 2	1	0
Channel 3	1	1

8. The interrupt vector is utilized by the CPU as the least significant 8 bits of a 16 bit pointer which provides the starting address of interrupt service subroutine for that particular channel. The upper 8 bits of this pointer are provided by the contents of the I Register in the CPU.

9. Students are referred to Fig. 4-1 which graphically represents how the pointer is concatenated and the service subroutine is called out.

10. A system convention requires that all the 16-bit addresses, which are prestored in the interrupt service subroutine table in the memory, should have their low-order byte placed in an even location in the memory. The high-order byte is then placed in the next highest location which will always be an odd-numbered address. Thus, the LSB of any interrupt vector will always be even as indicated by a 0 in the D0 position.

11. The last instruction in all interrupt service subroutines is RETI which initializes the daisy chain enable line IEO. This ensures proper control for handling priorities in the interrupt system. The CTC monitors the CPU data bus, decodes this instruction and executes it.

8-4 PACKAGE PIN ASSIGNMENTS AND FUNCTIONS

The CTC is contained in a standard 28-pin DIP package. The various pins and their respective functions, along with their designations are presented in convenient tabular form. Functionally, the pins can be divided into five groups as shown in Fig. 8-3. Note that some of the lines are shown as unidirectional and others as bidirectional. In the latter group, some of the lines may indicate signal flow in one direction and others in the other direction. The function tables clarify these with the following

arrow symbols:

CTC ⟶ Indicates signals flowing from the CTC
CTC ⟵ Indicates signals flowing into the CTC
CTC ⟷ Indicates bidirectional flow on the same line(s)

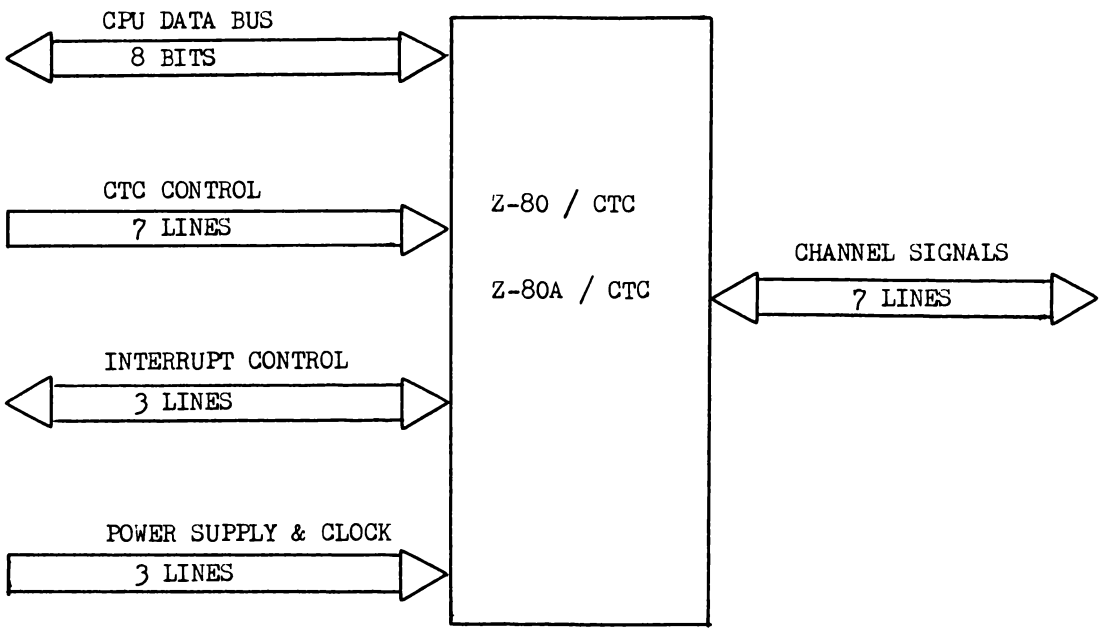


Fig. 8-3 Block diagram of CTC package pin groups.

Table 8-3 Data transfer bus group.

PIN DESIGNATIONS	SIGNAL FLOW DIRECTIONS	FUNCTIONS
D0-D7	CTC ⟷	8-bit, tristate, CPU data bus, is active high. It is used for data and commands transfer between the CTC and the CPU.

Table 8-4 Power supply and clock group.

PIN DESIGNATIONS	SIGNAL FLOW DIRECTIONS	FUNCTIONS
ø	CTC ◄——	Single-phase, TTL level, clock input.
+5 V	CTC ◄——	This is the single power supply of +5 ± 5 % volts that is required.
GND	CTC ◄——	This is the power supply ground.

Table 8-5 CTC control group.

PIN DESIGNATIONS	SIGNAL FLOW DIRECTIONS	FUNCTIONS
CS0 - CS1	CTC ◄——	2-bit binary code for selecting one of the 4 channels on the CTC chip during I/O operations. Generally connected to lines A0 - A1 of the CPU address bus.
$\overline{M1}$	CTC◄——	Machine Cycle One signal, active low, is a SYNC pulse from the CPU. When M1 and $\overline{RD}$ are simultaneously active, the CPU is fetching an instruction from the memory. When M1 and $\overline{IORQ}$ are simultaneously active, the CPU is acknowledging an interrupt. The CTC then places its interrupt vector on the CPU data bus provided it has the highest priority and an interrupt has been requested by a CTC channel.
$\overline{IORQ}$	CTC ◄——	I/O Request from CPU signal, active low, is used for transferring data between CPU and CTC as well as for sending channel control words to the CTC. The CTC write cycle is initated by the active status of $\overline{CE}$ and $\overline{IORQ}$ signals and inactive of $\overline{RD}$. (Note: CTC does not receive a specific external write signal but generates its own by inverting the $\overline{RD}$ signal). A CTC read cycle is initiated by simultaneous activation of $\overline{CE}$, $\overline{IORQ}$ and the $\overline{RD}$ signals and the contents of the Down Counter are transferred to the CPU on the data bus. When $\overline{IORQ}$ and M1 are simultaneously active, the CPU is acknowledging an interrupt and so the highest priority channel sends its vector to the CPU via the data bus.
$\overline{RD}$	CTC ◄——	Read cycle status, active low, signal when used simultaneously in conjunction with $\overline{CE}$ and $\overline{IORQ}$, it signals the CTC to transfer contents of the Down Counter to the CPU. During the write cycle, it is inactive while the other two are active. The CTC then accepts channel control word from the CPU or data transfers between CTC and the CPU.

Table 8-5 (Cont.) CTC control group.

PIN DESIGNATIONS	SIGNAL FLOW DIRECTIONS	FUNCTIONS
$\overline{CE}$	CTC ◄——	Chip enable, an active low signal that informs the CTC to transmit the contents of the Down Counter to the CPU during the Read cycle. During the Write cycle, it informs the CTC to accept either control words, time constant words or interrupt vectors from the CPU.
$\overline{RESET}$	CTC ◄——	Reset signal, active low. Stops the down counting process of all channels in the CTC and resets the interrupt enable bits in all control registers. This kills all interrupts generated by the CTC. Also, the zero count/time out and the $\overline{INT}$ output lines are put into the inactive states. The D0 - D7 bus lines go into the high-impedance states, and the IEO line is set to the same state as the IEI line.

Table 8-6 The interrupt control group.

PIN DESIGNATIONS	SIGNAL FLOW DIRECTIONS	FUNCTIONS
$\overline{INT}$	CTC ⟶	Interrupt request. Is active when the Down Counter of any channel in the CTC, that is programmed to enable interrupt, reaches the zero count.
IEI	CTC ⟵	Interrupt Enable In, active high. Used in the system-wide priority interrupt daisy chain. A high or active signal indicates that no other I/O device with higher priority is being currently serviced.
IEO	CTC ⟶	Interrupt Enable Out, active high. Also used in the priority interrupt daisy chain. It is active only when IEI is active and the CPU is not servicing an interrupt from any other CTC channel. When low, it blocks interrupt signals from lower priority I/O devices from interrupting while it is being serviced.

Table 8-7 Channel signals group.

PIN DESIGNATIONS	SIGNAL FLOW DIRECTIONS	FUNCTIONS
CLK/TRG_0 - CLK/TRG_3	CTC ◄——	External Clock/Timer Trigger, active either high or low as selected by the user. Four such lines are available for each independent channel. When used in the Counter Mode, the signal transition (i.e. either from high-to-low or low-to-high, as predefined), decrements the Down Counter. When used in the Timer Mode, the signal transition initiates the timing function.
ZC/TO_0 - ZC/TO_2	CTC ——►	This is the Zero Count/Time Out signal, from channels 0, 1 and 2 only, which is active high. When the Down Counter goes to zero, in either the Counter or the Timer Mode, this signal goes high.

Table 8-8 Pin number assignments.

PIN DESIGNATIONS	PIN NUMBERS	PIN DESIGNATIONS	PIN NUMBERS
CPU Data Bus		Channel Signals	
D0	25	CLK/TRG 0	23
D1	26	ZC/TO 0	7
D2	27	CLK/TRG 1	22
D3	28	ZC/TO 1	8
D4	1	CLK/TRG 2	21
D5	2	ZC/TO 2	9
D6	3	CLK/TRG 3	20
D7	4		
CTC Control		Interrupt Control	
CS0	18	$\overline{INT}$	12
CS1	19	IEI	13
$\overline{CE}$	16	IEO	11
$\overline{M1}$	14	Power Supply & Control	
$\overline{IORQ}$	10		
$\overline{RD}$	6	+5 volts	24
$\overline{RESET}$	17	GND	5
		ϕ	15

8-5 THE OPERATING MODES

8-5.1 Setting up the CTC Chip

1. When power is turned on, the CTC chip goes into an undefined and unpredictable state. Recall that the RESET signal from the CPU puts the CTC into a certain state which accomplishes the following.

   a. The interrupt-generating capabilities of the CTC are disabled,
   b. The Zero Count/Time Out and the INT output lines are put into the inactive states,
   c. The D0 - D7 bus lines go into the high-impedance states, and,
   d. The IEO line is set to the same state as the IEI line.

2. However, the RESET signal does not program the CTC chip. Before the desired channel can perform any timing or counting function, the CTC must be programmed with the following three words, in the given order:

   a. The Interrupt Vector - This 8-bit word is loaded into the Interrupt Control logic of the CTC. The function, format and operation of this word is explained in Sec. 8-3.3.

   b. The Channel Control Word - This 8-bit word is loaded into the Channel Control Register. The format of this word is explained in Sec. 8-3.2.

   c. The Time Constant Word - This 8-bit word is loaded into the Time Constant Register. It contains an integer whose value ranges from 1 to 256. When all 8 bits are zero, it is automatically interpreted as 256.

3. We will now explain the operation of each mode more in detail.

8-5.2 Operating in the Counter Mode

1. When set up in this mode, the function of the CTC chip is to count the number of signals or pulses inputted to the preselected CTC channel on the CLK/TRG line.

2. Each input signal then decrements the contents of the counter. Notice that the Time Constant Word had previously been loaded into the Down Counter.

3. When the Down Counter reaches 0, a pulse is outputted on the ZC/TO line of that particular channel. As previously explained, this output is available only for Channels 0, 1 and 2 (because of pin limitations on the package).

4. When the Down Counter reaches zero, an interrupt request signal (INT) is generated and sent to the CPU provided the channel has been programmed to do so by bit 7 of the Channel Control Word.

5. Notice that the incoming signal on the CKL/TRG line can be programmed (by means of bit D4 of the channel Control Word) to respond to either the positive-going or the negative-going edge of this pulse.

6. The CLK/TRG pulse inputs do not have to come into the CTC at regular timer intervals. They can, and do, come randomly. However, the Down Counter will not decrement until the next clock Ø following the CLK/TRG input.

7. When the Down Counter reaches zero, the Time Constant Register automatically reloads this counter with the 8-bit Time Constant Word.

8. If a new Time Constant Word is written into the Time Constant Register while the counter is in the decrementing process, the current count will be completed. The new time constant will be loaded only after the Down Counter has been decremented to zero.

8-5.3 Operating in the Timer Mode

1. When programmed in this mode, the function of the selected CTC channel is to provide a signal on the ZC/TO line when a certain pre-established time has elapsed.

2. The elapsed time can be programmed by the user and is the product of three factors. Two of these three factors are variable and directly controllable by the programmer. The elapsed time is given by the equation,

$$T_e = t_c \times PF \times TC$$

where, T_e is the elapsed time,

t_c is the system clock period,

PF is the Prescaler Factor, and

TC is the 8-bit integer time constant.

3. The operation of this mode is relatively simple. The system clock Ø is passed through two counters which divide the clock frequency at each stage. First, the Prescaler issues an output once every 16 or 256 input pulses of the clock Ø. This is then used as input clock for the Down Counter which is preloaded with a time constant.

4. If D3 of the channel control word is reset to 0, the CTC will start the timing operation automatically at the start of the CPU cycle following the I/O write cycle that loads the TC data word in the selected channel.

5. If D3 is set to 1, then the timing operation starts on the second succeeding

positive-going edge of Ø after it receives the trigger active edge on the CLK/TRG line of that channel.

6. If D2 of the channel control word is a 0, indicating that no time constant word follows, then the CTC starts the timing operation on the second succeeding positive-going edge of the clock Ø after the Timer Trigger edge following the control word inserted in the CTC.

7. When D7 of the channel control word is set to 1, the interrupt generating capabilities of the CTC are enabled and an interrupt request signal $\overline{(INT)}$ is sent to the CPU every time the Down Counter reaches a 0.

8. The Z-80 system now has the following three clock speeds available:

Z-80		2.5 MHz
Z-80A	-	4.0 MHz
Z-80B	-	6.- MHz

Using the TC range of 1 to 256 and the prescaler factor of 1 or 256, the elapsed time range available for each of the three systems is calculated from the equation in paragraph 2 above and given below:

For Z-80,	6.4 µs	to	26.214 ms
For Z-80A,	4.0 µs	to	16.834 ms
For Z-80B,	2.65 µs	to	10.88 ms

8-6 THE CTC TIMING

In this section we discuss the timing involved in the write cycle, the read cycle, and the counting and timing modes as well as the interrupt acknowledge and the return from the interrupt cycle. In the following timing charts, notice that the clock phases are shown as perfect rectangular waves. This is done only for convenience. Actually each clock pulse has its own rise and fall times. The rise and fall times of the other signals are shown as appropriate. The clock pulses are used in these charts simply for reference purposes and therefore they are shown as rectangular pulses.

8-6.1 The CTC Write Cycle

1. The write operation in the CTC refers to the operation involved in initialization of the chip. In other words, this relates to the acceptance of the channel control word, the time constant word and the Interrupt Vector sent by the CPU.

2. Note that the CTC does not have a separate write signal input. A write signal is internally generated when the $\overline{RD}$ signal goes high.

3. The timing chart for the write operation is shown in Fig. 8-4. During or before

clock period T1, the $\overline{RD}$ goes high. This prepares the CTC chip for the write cycle.

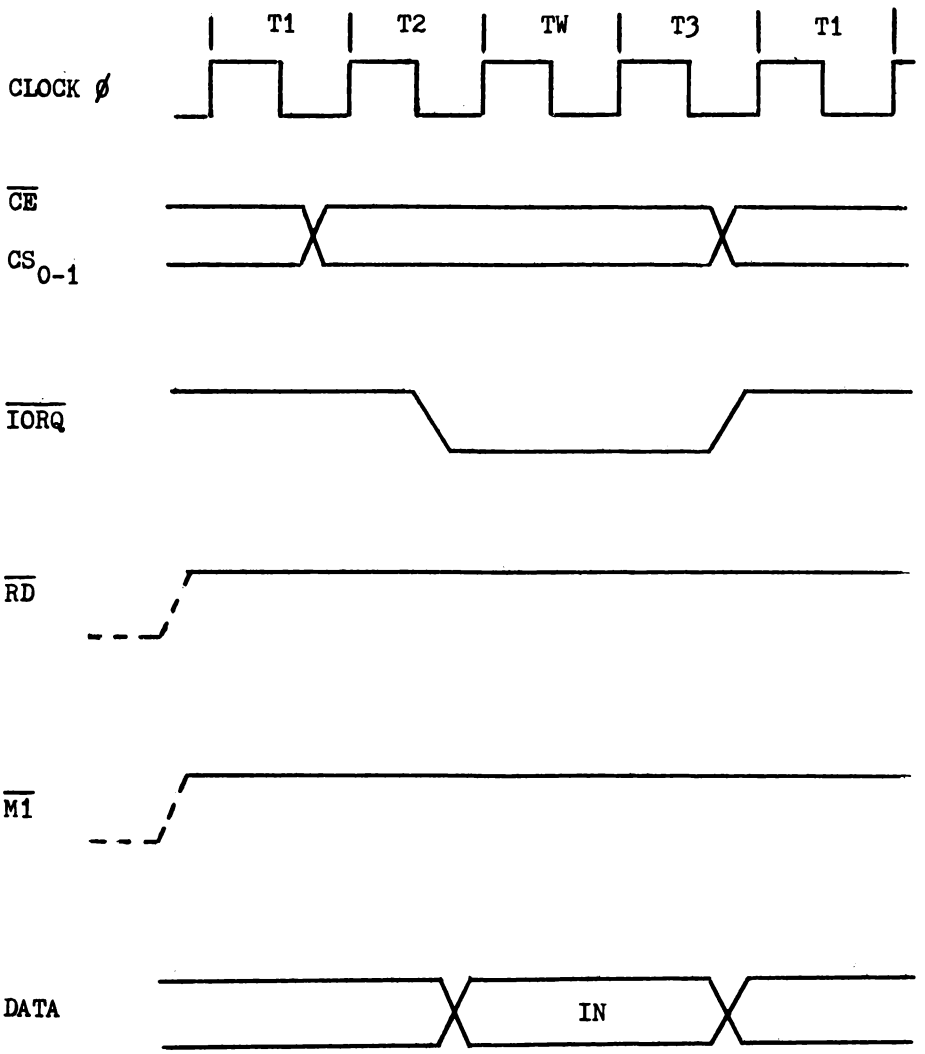


Fig. 8-4 CTC Write Cycle timing.

4. During clock period T2, the CPU initiates the write cycle by putting the chip enable, $\overline{CE}$, and the $\overline{IORQ}$ signals in the active or low state. Notice that during this time $\overline{M1}$ must be inactive, that is in the high state, otherwise it will be misinterpreted as an interrupt acknowledge cycle.

5. Also, during the T2 clock period, the 2-input channel select code appears on lines CS0 and CS1. This, of course, specifies which of the four available CTC channels is selected for that particular operation.

6. Then, during T2, the CPU puts the word to be written into the CTC chip on the CPU data bus lines D0 - D7.

7. The TW wait state is automatically inserted by the CPU. At the leading edge of clock Ø T3 the word outputted by the CPU is latched into the appropriate internal register of the CTC.

8-6.2 The CTC Read Cycle

1. The function of the read cycle involves reading of the current contents of the Down Counter by the CPU. This operation does not destroy or alter the contents of the Down Counter.

2. The CPU initiates the read cycle during the T2 clock period by both the $\overline{RD}$ and the $\overline{IORQ}$ signals going to the active or low states. Note that the $\overline{CE}$ is also active during this time.

3. At the same time, the 2-bit binary code CS0 and CS1 also appears and selects the appropriate channel.

4. Once again note that the $\overline{M1}$ signal is inactive (i.e. high) during this time, thereby ensuring that this cycle is not misinterpreted as an interrupt acknowledge cycle.

5. The contents of the Down Counter are then placed on the CPU data bus during the rising edge of the clock Ø T3 as shown in Fig. 8-5. Notice that this word is available on the data bus only during the T3 period.

6. The wait state TW is automatically inserted by the CPU.

8-6.3 The CTC Counter Mode

1. Figure 8-6 shows the timing of the CTC operation in the Counter Mode. The function of the CTC in this mode is to count the number of incoming signals on the CLK/TRG input and then output a signal on the ZC/TO line when the pre-established number of incoming signals have been counted by the Down Counter going to zero.

2. In the example shown in Fig. 8-6, the CTC is programmed to respond to the positive-going input signals on the CLK/TRG line. (Note: this is accomplished by a 1 bit in the D4 position of the channel control word).

3. Each incoming signal on the CLK/TRG line decrements the Down Counter. This is of course synchronized with the clock Ø.

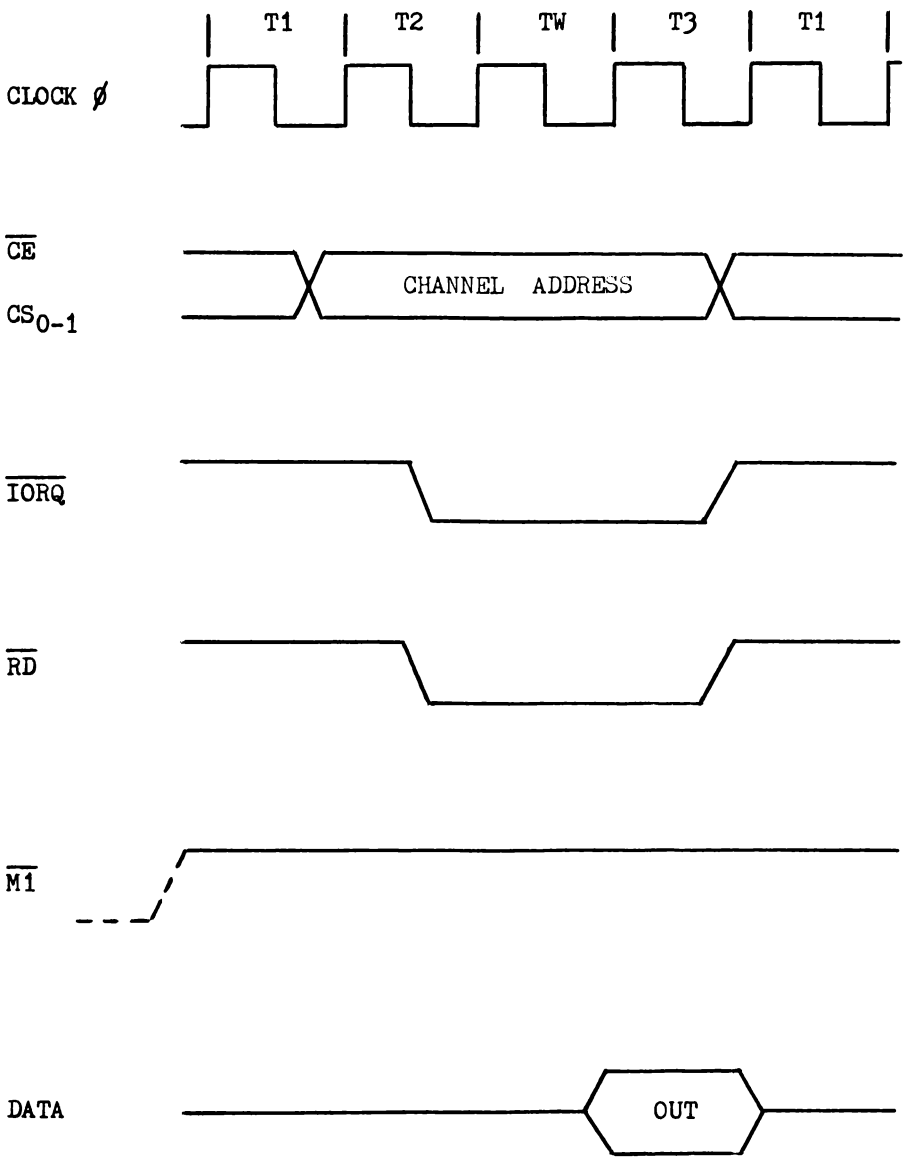


Fig. 8-5 CTC Read Cycle timing.

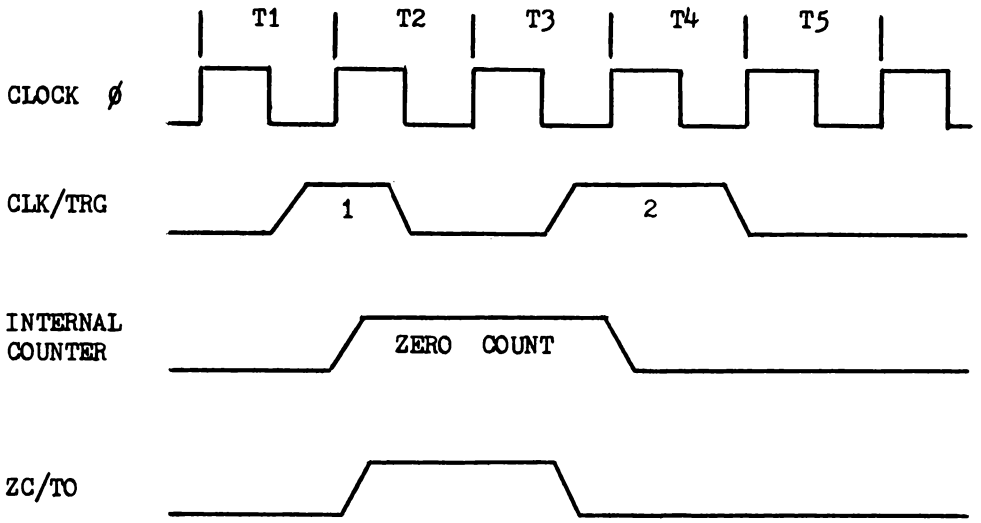


Fig. 8-6 CTC Counter Mode timing.

4. In Fig. 8-6 notice that the CLK/TRG pulse, identified as #1, is the last input which makes the Down Counter go from a 1 to a 0. At the rising edge of the next clock Ø, T2, the Down Counter goes to zero. Almost immediately a pulse is sent out on the ZC/TO line.

8-6.4 The CTC Timer Mode

1. Figure 8-7 shows the timing of the CTC operation in the Timer Mode. The function of the CTC in this mode is to provide a signal on the ZC/TO line when a certain, preprogrammed time inerval has elapsed after the timing operation has been started.

2. In this mode, a trigger pulse at the CLK/TRG input starts the process. The CTC could be triggered by either the positive-going or the negative-going edge depending upon how the D4 bit is programmed in the channel control word.

3. As seen in Fig. 8-7, the internal timing process starts on the leading edge of the next clock Ø (i.e. T2 in this case), since the trigger input can arrive at the CTC any time. In other words it is an asynchronous input.

4. When the programmed time has elapsed, as indicated by the Down Counter reaching 0, a pulse is sent out on its ZC/TO line, such as the one shown in Fig. 8-6 for the counter mode.

5. The pulse on the ZC/TO line is not shown in Fig. 8-7 since the elapsed time is a

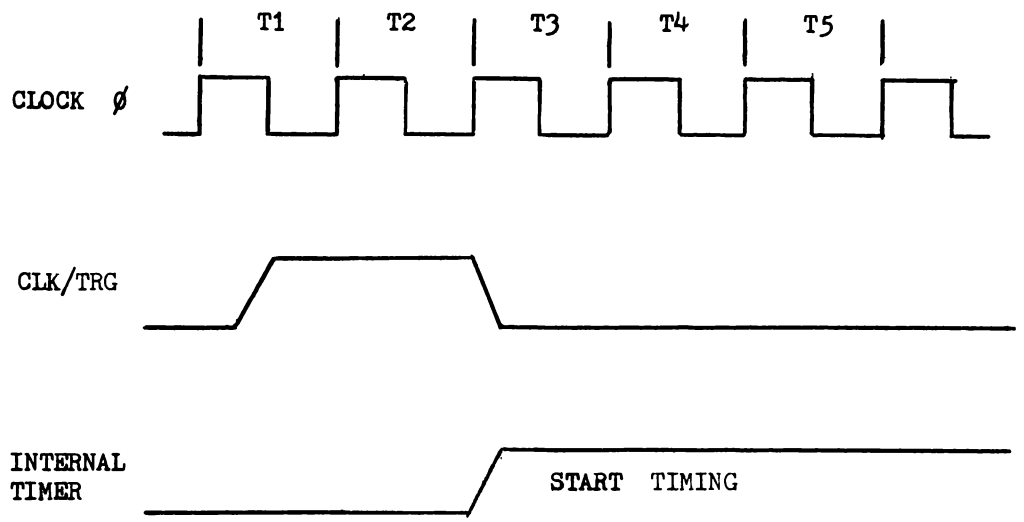


Fig. 8-7 CTC Timer Mode timing.

variable quantity which is different in each application.

8-6.5 Interrupt Acknowledge Cycle Timing

1. As we have previously discussed, the CTC has the capability of being programmed for the generation of an interrupt request signal $\overline{INT}$ every time the Down Counter reaches zero.

2. Fig. 8-8 shows the timing involved in the interrupt acknowledge cycle.

3. Sometime after receiving an $\overline{INT}$ signal, the CPU acknowledges the request by activating the $\overline{M1}$ and $\overline{IORQ}$ lines.

4. During this time the $\overline{RD}$ line is deactivated (i.e. is high) otherwise it might be interpreted as an instruction fetch cycle.

5. In Fig. 8-8 notice that the $\overline{IORQ}$ signal is activated about two clock periods after the $\overline{M1}$ is activated. When $\overline{M1}$ is active, the channels cannot change their interrupt status thus ensuring that the daisy chain lines have ample time to stabilize.

6. During this time interval, the interrupt logic in the CTC chip will determine the highest priority channel requesting the interrupt.

7. The two wait states, TW1 and TW2, are automatically inserted by the CPU to enable

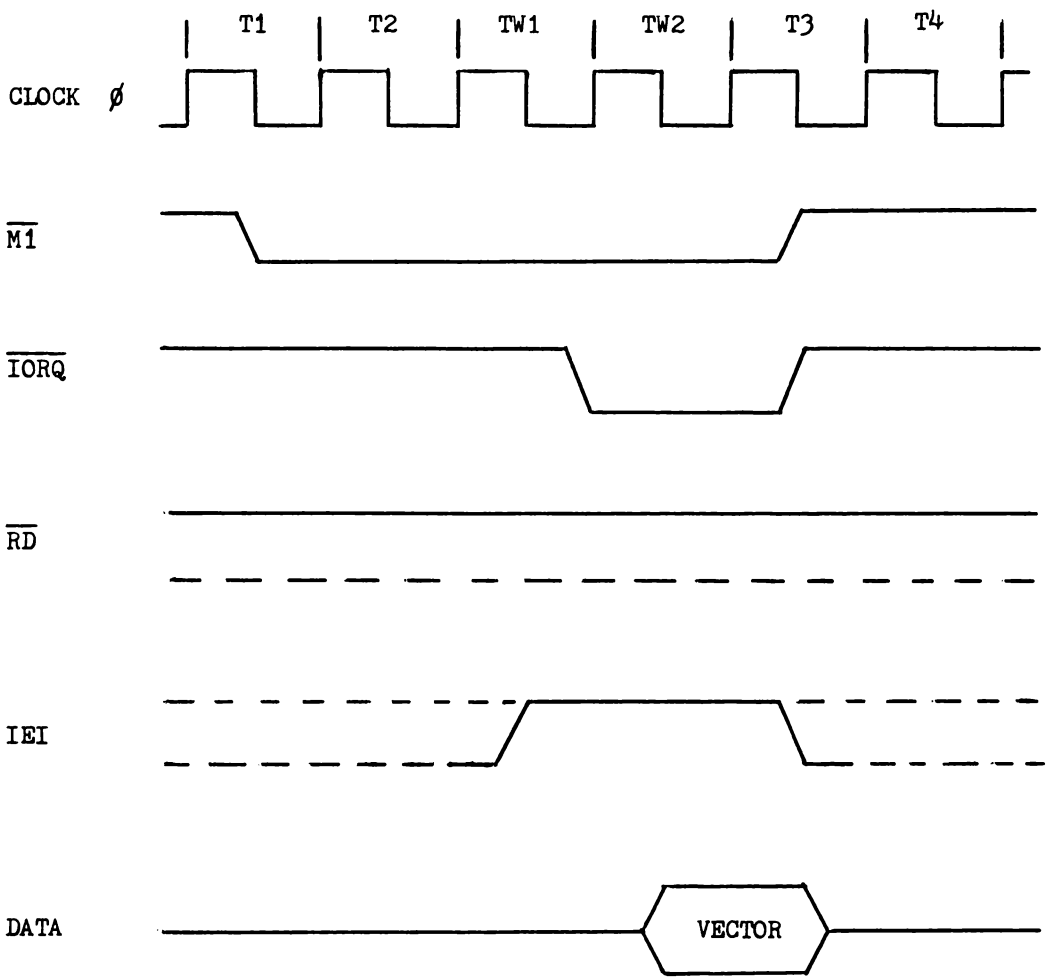


Fig. 8-8 Interrupt Acknowledge cycle timing.

the daisy chain lines to stabilize.

8. Then, during the TW1 clock period, if the CTC IEI input is put in the high state by the external device (thereby indicating that it has the highest priority in the daisy chain), the CTC places the appropriate Interrupt Vector on the CPU data bus since both $\overline{IORQ}$ and IEI are active.

8-6.6 Return from Interrupt Timing

1. At the end of each interrupt service subroutine, the RETI instruction is used. This instruction sets up the daisy chain for proper control of the interrupt priority handling logic for the daisy chain.

2. Fig. 8-9 shows the timing involved with the execution of the RETI instruction. The 2-byte RETI code is internally decoded by the CTC.

3. The EDH OP code is placed on the data bus when $\overline{M1}$ and $\overline{RD}$ are simultaneously active. The IEI line is then activated and if the following OP code is 4DH, then the I/O device which is being serviced will be re-initialized. After transfer of the 4DH OP code, the IEO line is activated.

8-7 REVIEW QUESTIONS

8-1 The statements below refer to the CTC chip. Indicate true or false.

a. ___ The 4 channels on the CTC are independent of each other.

b. ___ When used in the counter mode, the CTC counts the number of clock pulses inputted to it.

c. ___ When used in the timer mode, the CTC generates time intervals by dividing the system clock frequency.

d. ___ When used in either counter or timer mode, channel 3 has the highest priority and channel 0 the lowest.

e. ___ The CTC can be directly interfaced with the Z-80/SIO for generating the required baud rate.

8-2 Answer yes or no for the CTC chip.

a. ___ When used in the timer mode, can teh CPU read the number of counts still to go before zero?

b. ___ In either counter or timer mode, when the Down Counter reaches zero can it be reloaded automatically?

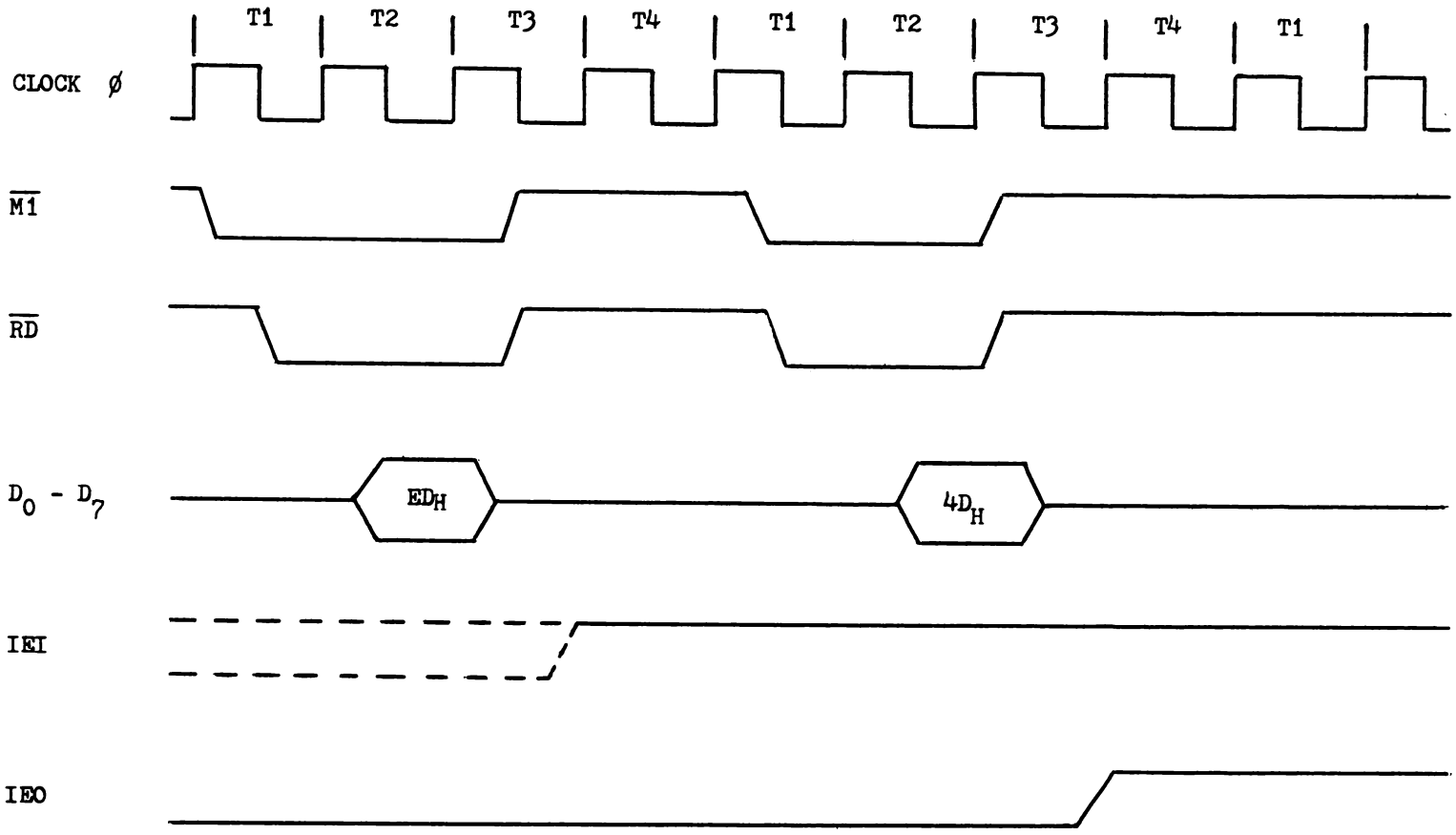


Fig. 8-9 Return from Interrupt timing.

c. ___ Is it possible to initiate timer operation by means of a negative-going trigger?

8-3 Refer to the generalized CTC block diagram of Fig. 8-1 and indicate true or false.

a. ___ The 4 channels can be connected into 4 consecutive slots of the Z-80 daisy chain.

b. ___ A single interrupt vector is used as a common service subroutine starting address for all 4 channels.

c. ___ The interrupt control lines between the CTC and the CPU are all bidirectional lines.

d. ___ The read/write, chip enable and the reset functions are handled by the block labelled Interrupt Control Logic.

e. ___ During the interrupt service subroutine execution, the IEO signal is held high.

f. ___ During the interrupt service subroutine execution, the IEO signal is used to inhibit interrupt requests from I/O devices with lower priorities.

8-4 The following statements relate to the Channel Logic block diagram of Fig. 8-2. Fill in the blanks.

a. The Channel Control Register receives _____ bit control word from the _____. (6; 8; 16; CPU; I/O devices; $Channel_{0-3}$).

b. One of the 4 channels is selected by a _____ bit code from the CPU on the _____ pins. (2; 4; 6; 8; CS_{0-1}; ZC/TO; CLK/TRG)

c. The CTC pins CS_0 and CS_1 are usually connected to the _____ and _____ lines from the CPU. (D0; D1; A0; A1; ZC/TO_0; ZC/TO_1)

d. Bit _____ of the channel control word selects the prescaler value of _____ or _____. (D3; D5; D7; 8; 16; 64; 128; 256)

8-5 Indicate true of false for the channel logic block diagram of Fig. 8-2.

a. ___ The Prescaler is used only in the Counter Mode.

b. ___ The divided output of the Prescaler is used as the clock input to the Down Counter.

c. ___ The Time Constant Register is used in both the counter and the timer operation modes.

d. ___ The time constant is an integer whose value ranges from 0 to 255.

e. ___ The time constant is loaded into the Down Counter when the CTC is first initialized.

8-6 Fill in the blanks for the Down Counter operation.

a. The Down Counter is _____ when used in the counter mode. (Incremented; decremented)

b. The Down Counter is _____ when used in the timer mode. (Incremented; decremented)

c. If _____ of the channel control word is a _____ , the Down Counter is triggered automatically. (D3; D6; 0; 1)

d. The Down Counter is triggered by the active edge of the CLK/TRG pulse only when bit _____ of the channel control word is a _____ . (D1; D2; D3; D4; 0; 1)

8-7 The following statements relate to the Interrupt Control Logic and Protocol. Indicate true or false.

a. ___ For the Down Counter to generate an interrupt request, the CPU must be programmed for Mode 2 operation.

b. ___ Higher priority channels can override the lower priority channels.

c. ___ If the IEI signal is active, the CTC places its interrupt vector on the data bus.

d. ___ Bits D3 - D7 of the interrupt vector are preloaded during the CTC initialization.

e. ___ The interrupt vector is utilized by the CPU as the least significant 8 bits of a 16-bit pointer.

8-8 The statements below apply to the 16-bit pointer which provides the starting address of the interrupt service subroutine. Indicate true or false.

a. ___ The upper 8 bits of the pointer are provided by the R Register of the Z-80/CPU.

b. ___ 16-bit starting addresses of the interrupt service subroutines are prestored in a table in the memory.

c. ___ The low-order byte of the address is placed in an even location in the memory.

d. ___ The LSB of the interrupt vector will always be a 0.

e. ___ The first instruction in all interrupt service subroutines is always the RETI instruction.

8-9 Fill in the blanks in the following statements.

a. The Machine Cycle One signal is active _____, (High; low)

b. The CPU is fetching an instruction from the memory when $\overline{M1}$ and _____ are simultaneously active. ($\overline{RD}$; $\overline{IORQ}$; $\overline{CE}$)

c. The CPU is acknowledging an interrupt when $\overline{M1}$ and _____ are simultaneously active. ($\overline{CE}$; $\overline{IORQ}$: $\overline{IEO}$; $\overline{IEI}$; RESET)

d. Assuming that it has the highest priority, the CTC places its interrupt vector on the CPU data bus after _____ and _____ are simultaneously active.

($\overline{RD}$; $\overline{CE}$; $\overline{RESET}$; $\overline{IORQ}$; $\overline{IEI}$; $\overline{M1}$)

8-10 _____ Name the signal that the CPU sends to the CTC for sending the channel control word to the CTC as well as for transferring data between the CPU and the CTC.

8-11 The CTC write cycle is initiated by the active status of _____ and _____ signals and the inactive status of the signal.
($\overline{CE}$; $\overline{RD}$; M1; $\overline{IORQ}$)

8-12 To transfer the contents of the Down Counter to the CPU, the following signals must be simultaneously active: (Circle all that apply)

a. $\overline{CE}$
b. $\overline{RD}$
c. $\overline{M1}$
d. $\overline{IORQ}$
e. IEO
f. IEI

8-13 Circle those signals that are active during the CTC write cycle.

a. $\overline{CE}$
b. $\overline{RD}$
c. $\overline{M1}$
d. $\overline{IORQ}$
e. IEO
f. IEI

8-14 Which of the following signal(s) will stop the down counting process of all channels in the CTC?

a. $\overline{RESET}$
b. $\overline{RD}$
c. $\overline{IORQ}$
d. $\overline{M1}$

8-15 Which of the following signal(s) will reset the interrupt enable bits in all control registers? Circle the one(s) that apply.

a. $\overline{RD}$
b. $\overline{M1}$
c. $\overline{RESET}$
d. $\overline{IORQ}$

8-16 Channel 1 is programmed to enable interrupt. Which of the following signal(s) will be issued by the CTC when the Down Counter reaches zero?

a. $\overline{\text{RESET}}$ c. $\overline{\text{INT}}$
b. $\overline{\text{IEI}}$ d. $\overline{\text{IORQ}}$

8-17 The following statements apply to the signals that are communicated between the I/O devices and the various CTC channels. Indicate true or false.

a. ___ The CLK/TRG signal can be selected to be active either high or low.

b. ___ When used in the timer mode, each predefined signal on the CLK/TRG pin decrements the Down Counter.

c. ___ When used in the counter mode, each predefined signal on the CLK/TRG pin initiates the timing function.

d. ___ When the Down Counter goes to zero, the ZC/TO output signal goes high.

8-18 Does the RESET signal program the CTC chip? Check one.

a. ___ Yes.

b. ___ No.

8-19 Indicate true or false for the CTC Counter mode oepration.

a. ___ The CLK/TRG inputs do not have to come to the CTC at regular time intervals.

b. ___ The new time constant will be loaded only after the Down Counter has decremented to 0.

c. ___ When the Down Counter of channel 3 reaches 0, a pulse is outputted on the ZC/TO_3 line.

d. ___ When the Down Counter reaches 0, the Time Constant Register reloads the Down Counter with the time constant word.

8-20 The following statements relate to the CTC Timer Mode operation. Fill in the blanks.

a. The elapsed time is given by the _____ of the system clock period, the Prescaler Factor, and the Time Constant. (Sum; product)

b. The system clock Ø is passed through two _____ which _____ the frequency at each stage. (Register; counters; adds; divides; multiplies)

c. The output of the _____ is used as a input clock for the _____. (Prescaler; Clock Ø; Down Counter; Time Constant Register)

9 PROGRAMMING THE CTC CHIP

CHAPTER OBJECTIVES

The objectives of this chapter are as follows:

1. To introduce the concept of customizing the CTC chip for specific applications involving counting and timing functions.
2. To describe the function and format of the Channel Control Word.
3. To discuss the function of the Time Constant Register and the format of the Time constant Data Word.
4. To present and discuss the formation of the 16-bit pointer for the starting address of the interrupt service subroutine when the CTC is programmed for Mode 2 interrupt response.
5. To set up and program the CTC chip for a typical operation in the counter mode by means of an example.
6. To demonstrate how the CTC can be programmed for the timing operation by means of a typical example.

TEXTBOOK REFERENCES

Counter and Timer circuits are not covered in the main hardcover textbook because it was felt that these subjects are adequately covered in several excellent books on digital logic. One book that this author has found particularly useful is "Practical Digital Design Using IC's" by Joseph D. Greenfield, a John Wiley publication.

9-1 INTRODUCTION

The CTC is a programmable interface chip which, under program control, can be customized to perform functions for specific applications within certain limits. Customizing the

CTC requires that the chip be set up first by means of an initialization program which in this case consists of only three words. In this chapter we describe the initialization process and the words involved in such a program. Also, examples are included to show how this chip can be programmed for some typical counting and timing applications.

9-2 THE PROGRAMMING WORDS

In this section we describe and discuss the formats of the various initialization words. These words are presented in the sequence in which they would be included in the initialization program.

9-2.1 The Interrupt Vector

For interrupt operations with the CTC, the Z-80/CPU must be programmed for Mode 2 interrupt responses. This mode, which is the most versatile and powerful interrupt response, was described in Chapter 4. Students are referred to Sec. 4-4.6.3 and Fig. 4-1 for a review of this mode.

In Mode 2 interrupt response, the CPU receives an interrupt request from one of the CTC channels and then responds with an interrupt acknowledge. The true or effective starting address of the appropriate interrupt service subroutine is stored in a starting address table in the memory. This address is called out by a 16-bit address pointer, using the indirect addressing mode. This 16-bit pointer is formed by concatenating the prestored contents of the I Register in the CPU, which forms the upper 8 bits, and the 8-bit vector provided by the CTC, which forms the lower 8 bits. Fig. 9-1 shows the formation of this pointer. Notice that bit 0 of this vector is always 0.

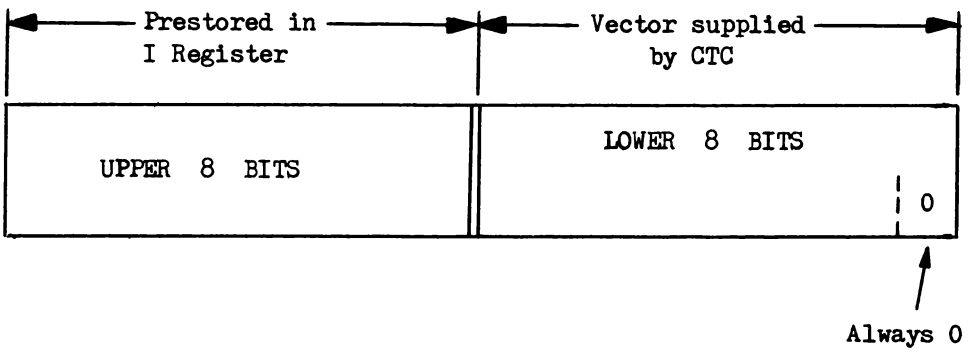


Fig. 9-1 Pointer for interrupt subroutine starting address.

The 0 in bit 0 positon of the vector identifies it as the interrupt vector. As we shall see in Sec. 9-2.2, a 1 bit in bit 0 position will identify it as a Channel Control word. The 2-byte effective starting address of the subroutine is always stored in the table in the memory with the low-order byte first and the high order byte in the next addressed

location. Since bit 0 of the interrupt vector is always a 0, this means that the 16-bit pointer will always point to an even address in the starting address table.

During initialization of the CTC, the interrupt vector is loaded first. The format of this 8-bit word was previously presented in Sec. 8-3.3. For convenience it is repeated here in Fig. 9-2. In this word, the upper five bits, D3 - D5, are supplied by the programmer. Bit D0 is always a 0 (as previously explained). The two bits for positions D1 and D2 are automatically supplied by the interrupting channel according to the codes in Table 8-2. These codes are also repeated for convenience in Table 9-1.

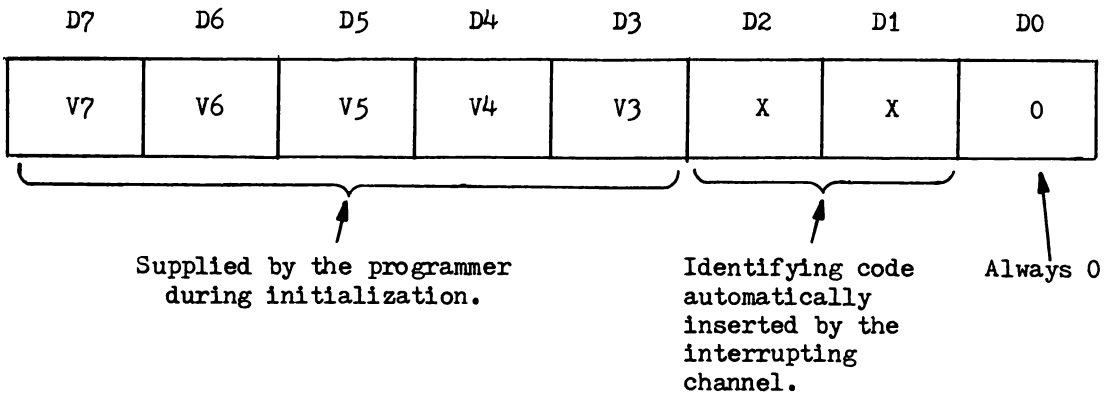


Fig. 9-2 Interrupt Vector format.

Table 9-1 Interrupt channel identification codes.

	D2	D1
Channel 0	0	0
Channel 1	0	1
Channel 2	1	0
Channel 3	1	1

During initialization, the CPU writes the interrupt vector into the CTC chip by writing into the I/O port address corresponding to channel 0 of the CTC. This means that bits D2, D1 and D0 must all be 0 during the writing in process. Also remember that Channel 0 has the highest priority and Channel 3 the lowest.

9-2.2 The Channel Control Word

The next word in the initialization program is the channel control word which is loaded in the 8-bit Channel Control Register (See Fig. 8-2.) The format of the channel control word, and the function of each bit in that word were described in Sec. 8-3.2 in reasonable detail, and so will not be repeated here. However, for convenience, the format of the channel control word is repeated in Fig. 9-3 and the meanings of each of the bits is also given following Fig. 9-3. Note that bit position D0 is always a 1 for the channel control word.

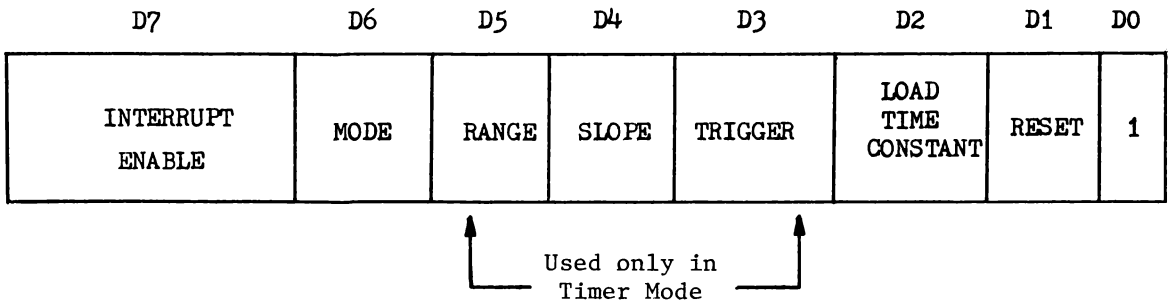


Fig. 9-3 Channel Control Word format.

D0 - Word Identifier
Always 1 for Channel Control Word

D1 - Reset
1 Software reset
0 Continued operation

D2 - Load Time Constant
1 Time constant follows
0 No time constant follows

D3 - Trigger
1 Clock/trigger pulse starts the timer
0 Automatic trigger when time constant is loaded

D4 - Slope
1 Selects positive-going edge of clock/trigger pulse
0 Selects negative-going edge of clock/trigger pulse

D5 - Range
1 Indicates prescaled value of 256
0 Indicates prescaled value of 16

D6 - Mode
1 Indicates Counter mode
0 Indicates Timer mode

D7 - Interrupt Enable
1 Enables interrupt
0 Disables interrupt

The following additional explanations are intended to aid students in understanding and composing the Channel Control Word for the initialization program.

1. When bit D1 is set, i.e. it is a 1, the channel involved stops the timing or the counting process. In other words, the current operation of the channel is terminated. However, if both bits D1 and D2 are concurrently 1, the channel will resume operation when a time constant word is loaded. If D1 is a 0, the channel continues with the current operation.

2. A 1 in the D2 bit position informs the logic in the selected CTC channel that the next word is a time constant word which should be loaded into the Time Constant Register (Fig. 8-2). A 0 in this bit position indicates that no time constant is to follow and that this channel control word is written to update the status of the channel that is currently in operation.

3. Bit position D3 is active only when CTC is operating in the timer mode. When this bit is set, an external trigger pulse starts the timing operation after the leading edge of T2 of the machine cycle following the one that loads the time constant word as shown in the timing chart of Fig. 8-7.

4. A 0 in the D7 position disables the interrupt generating capability of the selected CTC channel. On the other hand, a 1 in this position results in the generation of an interrupt signal on the $\overline{INT}$ line every time the down counter reaches the zero condition. This feature is applicable in both the timer and the counter modes. Is of course possible to write an updated channel control word while the particular channel is already in operation. In such a case, if the previous channel control word had a 0 in the D7 position, then a 1 in the updated word will not pre-empt the previously defined condition and generate an interrupt when the down counter reaches the zero count condition. The conditions specified by the updated channel control word become operative only after completion of the process defined by the previous channel control word.

9-2.3 The Time Constant Word

The format of the time control word, which is loaded into the Time Constant Register of Fig. 8-2, is shown in Fig. 9-4. It is an 8-bit word whose integer value ranges from 1 to 256. If all 8 bits are 1s, its value is 255. If all 8 bits are 0s, its value is interpreted as 256.

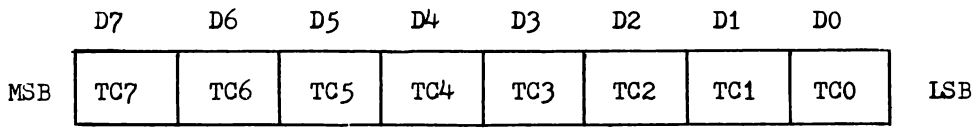


Fig. 9-4 Time Constant Word format.

The logic of the CTC is designed such that a channel will not operate, in either the counter or the timer mode, unless a time constant word is previously loaded in the Time Constant Register of that channel. This means that in the initialization program, bit position D2 of the channel control word must contain a 1. If it is a 0, it only means that the channel is already in operation and that the new channel control word is only an update. Note that the time constant word can be written into the Time Constant Register at any time without interfering with the operation of the down counter which may currently be in the process of counting down. The down counter will continue decrementing to zero and the new time constant word will be loaded into it only after the count has reached zero.

9-3 PROGRAMMING FOR COUNTER APPLICATIONS

Example 9-1

The interrupt page address register in the CPU (the I Register) is preloaded with $1D_H$ and the CPU is preprogrammed for Mode 2 interrupt response. The starting address table is stored from memory location $1D18_H$ to $1D1F_H$. Channel 0 is operated in the counter mode and interrupt logic is enabled. The channel is operated with the initial time constant of $3E_H$ and the down counter is decremented on the positive-going edge of the input trigger pulse. Channels 1, 2 and 3 are inactive. Write the program words required to initialize the CRC for this problem in both hexadecimal and machine codes and explain what you did and why.

Solution

The first initialization word that must be loaded in the CTC chip is the interrupt vector. Since the address of the starting address table, stored in the memory, starts with $1D18_H$,

the pointer for the channel 0 interrupt service subroutine is $1D18_H$. Since $1D_H$ is preloaded into the I register of the CPU, the interrupt vector from the CTC must supply the lower 8 bits of the pointer as shown in Fig. 9-1. Therefore the interrupt vector for channel 0 must be 18_H. The machine code for this is shown below.

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	1	1	0	0	0

As explained in Sec. 9-2.1, bit D) is a 0. The 5 high-order bits (i.e. D7 - D3) are supplied by the programmer. Although bits D2 and D1 are given as 0 and 0 respectively, the interrupting channel will automatically insert its proper identifying code in these two bit positions as shown in Table 9-1.

The channel control word is the next word to be loaded.

1. Since the interrupt is enabled, bit D7 is a 1.
2. Bit D6 is a 1 because we are operating in the counter mode.
3. Since bits D5 and D3 are both applicable only in the timer mode, the bits in these two positions are not relevant in this case. We will, therefore, arbitrarily, insert two 0s here.
4. Bit D4 is a 1 because the problem statement calls out for the down counter to be triggered on the positive-going trigger input pulse.
5. Bit D2 is a 1 because a time constant word follows the channel control word.
6. We put a 1 in bit D1 position. This terminates the current operation of channel 0 but since D2 is also a 1, channel 0 will resume operation when the time constant word is loaded.
7. Bit D0 is always a 1 and it identifies this word as the channel control word.

The complete channel control word is shown below:

D7	D6	D5	D4	D3	D2	D1	D0
1	1	0	1	0	1	1	1

The next word to be loaded is the time constant word which is given in the problem

statement as $3E_H$. The machine code for this is,

D7	D6	D5	D4	D3	D2	D1	D0
0	0	1	1	1	1	1	0

The three CTC initialization words for this problem are shown in Table 9-2 with both the machine codes and the hexadecimal codes.

Table 9-2 CTC initialization words for example 9-1.

	HEX CODES	D7	D6	D5	D4	D3	D2	D1	D0
Interrupt Vector	18	0	0	0	1	1	0	0	0
Channel Control Word	D7	1	1	0	1	0	1	1	1
Time Constant Word	3E	0	0	1	1	1	1	1	0

Example 9-2

Refer to the problem statement of Example 9-1. Using this same data, construct the three CTC initialization words for channel 1 of the CTC but with the following two exceptions:

1. The down counter must be triggered on the negative-going edge of the input trigger pulse.

2. Channel 1 must generate an interrupt request when 254 input trigger pulses have been received.

Additional explanations need not be given. Give both the machine and the hexadecimal codes in tabular form similar to Table 9-2.

Solution

Table 9-3 CTC initialization words for example 9-2

	HEX CODES	D7	D6	D5	D4	D3	D2	D1	D0
Interrupt Vector	18	0	0	0	1	1	0	0	0
Channel Control Word	C7	1	1	0	0	0	1	1	1
Time Constant Word	FE	1	1	1	1	1	1	1	0

9-4 PROGRAMMING FOR TIMER APPLICATIONS

Example 9-3

Design the three initialization words for operating the CTC in the timer mode for the following conditions:

1. Channels 0 and 1 are previously programmed for operation in the counter mode.
2. Channel 2 is to be programmed for the timer mode and channel 3 is not used.
3. The I Register in the Z-80/CPU has been preloaded with $A4_H$.
4. Memory locations $A408_H$ through $A40F_H$ are reserved for storing the interrupt service subroutine starting addresses.
5. A prescaled value of 16 is used.
6. The time constant to be used is 23_H.
7. The interrupt vector is 08_H.
8. The interrupt is enabled.
9. The down counter is triggered on the positive-going edge of the clock timer input pulse.

Give both the hexadecimal and the machine codes for each word in tabular form and explain the procedure used as appropriate.

Solution

1. We program the interrupt vector for channel 0. When channel 2 generates an interrupt, it will automatically insert the appropriate channel identifier code (Table 9-1) in bit positions D2 and D1.

2. Thus, the interrupt vector is 08_H = 00001000.

3. It is assumed that the software reset is set and so bit position D1 is a 1.

4. Since a prescaled value of 16 is indicated, a 0 is used in bit position D5.

5. A 1 is used in D4 because a positive-going edge of the input pulse is used for triggering the timer.

6. A 1 is used for D3 because a trigger pulse is used for starting the timer.

7. Since a time constant word follows, a 1 is used in the D2 bit position.

8. The three set-up words are shown in Table 9-4.

Table 9-4 CTC initialization words for example 9-3.

	HEX CODES	D7	D6	D5	D4	D3	D2	D1	D0
Interrupt Vector	08	0	0	0	0	1	0	0	0
Channel Control Word	9F	1	0	0	1	1	1	1	1
Time Constant Word	23	0	0	1	0	0	0	1	1

Example 9-4

Design an initialization program for the CTC to operate in the timer mode for the following conditions and present both the hexidecimal and the machine codes in tabular form. Explanations may be inserted as appropriate.

1. The interrupt is enabled.

2. The clock is triggered on the negative-going edge of the input pulse.
3. Software reset is set.
4. A 2.5 MHz clock is used.
5. The interrupt vector is EO_H.
6. Channel 1 is to be programmed for this operation.
7. A time-out pulse must be outputted by the timer every 1.536 ms.
8. The I Register in the CPU is preloaded with 44_H.
9. The interrupt subroutine starting address table is stored in memory locations $44EO_H$ to $44E7_H$.
10. A prescaler value of either 16 or 256 may be used.
11. A time constant of either 08_H or OF_H may be used.

Solution

In this problem there are two variables which are unknown, namely, the prescaler factor (PF) and the time constant integer (TC). Since the elapsed time (T_e) is 1.536 ms, we use the formula given in Sec. 8-5.3 to arrive at the values for PF and TC. By trial and error the values are found to be,

$$PF = 256 \text{ and } TC = OF_H.$$

Since the Z-80 system uses a 2.5 MHz clock, the system clock period is

$$1/2.5 \times 10^{-6} = 0.4 \times 10^{-6}$$

Then, the elapsed time between the time-out signals is given by,

$$T_e = t_c \times PF \times TC$$

$$= 0.4 \times 10^{-6} \times 256 \times 15$$

$$= 1536 \times 10^{-6}$$

$$= 1.536 \times 10^{-3} = 1.536 \text{ ms}$$

The initialization program is given in table 9-5.

Table 9-5 CTC initialization words for example 9-4.

	HEX CODES	D7	D6	D5	D4	D3	D2	D1	D0
Interrupt Vector	E0	1	1	1	0	0	0	0	0
Channel Control Word	AF	1	0	1	0	1	1	1	1
Time Constant Word	0F	0	0	0	0	1	1	1	1

9-5 REVIEW QUESTIONS

9-1 For interrupt operations with the CTC, the Z-80/CPU must be programmed for interrupt responses in

a. Mode 0
b. Mode 2
c. Mode 1

9-2 The starting addresses of the interrupt service subroutine is called out by using the (circle all that apply):

a. Register addressing mode.
b. Extended addressing mode.
c. Indirect addressing mode.
d. Implied addressing mode.

9-3 The following statements relate to the starting address of the interrupt service subroutine. Indicate true or false.

a. ___ The effective subroutine starting addresses are stored in a table in the memory.

b. ___ An 8-bit address pointer is used for calling out the starting address.

c. ___ The upper 8 bits of the address pointer are contained in the R Register in the CPU.

d. ___ The lower 8 bits of the address pointer are supplied by the CTC.

9-4 Bit 0 of the CTC interrupt vector is always a

a. 0
b. 1

9-5 The 16-bit effective starting address of the interrupt service subroutine is always stored in the memory table with the _____ order byte first, followed by the _____ order byte in the next addressed location. (Low; high)

9-6 The 16-bit address pointer always points to an _____ address in the starting address table. (Odd; even)

9-7 In the interrupt vector, the code for the following bit position(s) is/are automatically supplied by the interrupting channel:

a. D0
b. D0 and D1
c. D1 and D2
d. D0 and D2
e. D0 and D3 through D7
f. D3 through D7

9-8 The channel with the highest priority in the CTC is:

a. Channel 2
b. Channel 3
c. Channel 1
d. Channel 0

9-9 The following statements apply to the bits in the Channel Control Word. Indicate true or false.

a. ___ A 1 in the D2 bit position indicates that a time constant follows:

b. ___ A 1 in the D1 bit position indicates that software reset is enabled.

c. ___ The timer is started by the clock/trigger pulse if there is a 1 in the D3 position.

d. ___ A 0 in the D5 bit position indicates a prescaled value of 16.

9-10 The I Register in the CPU is preloaded with $3D_H$ and channel 2 is operated in the counter mode. The 16-bit address pointer for the starting address of the interrupt service subroutine is:

a. $3D01_H$
b. $3D04_H$
c. $3DAB_H$
d. $3DD9_H$

9-11 Channels 0 and 1 are operated in the counter mode and channels 2 and 3 are inoperative. The preloaded time constant word for both channels is 29_H. The channel control word for channel 0 is CF_H and for channel 1 it is $D7_H$. Indicate true or false.

a. ___ In channel 0, the down counter is triggered automatically when the time constant word is loaded.

b. ___ In channel 1, the negative-going edge of the clock/trigger pulse starts the down counter.

c. ___ Software reset is set for channel 1.

d. ___ In channel 0, the positive-going edge of the clock/trigger pulse starts the down counter.

e. ___ In channel 0, a clock/trigger pulse starts the down counter.

9-12 Is it possible to write an updated Channel Control Word while the particular channel is already in operation?

a. Yes
b. No

9-13 When there is a 1 in the D7 position of the Channel Control Word, every time the down counter reaches the zero condition, a signal is generated in one or more of the following signal line(s):

a. IEI
b. IEO
c. $\overline{INT}$
d. Clock/trigger pulse line

9-14 The word statements relate to the situation where the previous channel control word had a 0 in the D7 bit position and the current channel control word has a 1 in the D7 bit position. Indicate true or false.

a. ___ A 0 in the D2 position indicates that the current channel control word updates the status of the channel currently in operation.

b. ___ A 0 in the D2 position indicates that a time constant word does not follow the channel control word.

c. ___ The current channel control word pre-empts the previously defined condition and generates an interrupt when the down counter reaches zero.

d. ___ The conditions specified by the current channel control word become operative immediately as soon as this word is loaded.

9-15 Channels 0 and 1 are operated in the timer mode and channels 2 and 3 are inactive. The I Register is preloaded with 10100011. The CTC interrupt vector is 00111010. The following statements relate to the address pointer for the starting address table stored in the memory. Indicate true or false.

a. ___ The address pointer for channel 0 is $A33A_H$

b. ___ The address pointer for channel 1 is $A33A_H$

c. ___ The address pointer for channel 0 is $A338_H$

d. ___ The address pointer for channel 1 is $A3A3_H$

e. ___ The address pointer for channel 1 is $A33A_H$

f. ___ The address pointer for channel 1 is $A338_H$

9-16 Channel 2 of the CTC is operated in the timer mode. Which of the following pair of hexadecimal codes is appropriate for this operation? The first code in each pair is the address pointer and the second code is the channel control word.

a. BB_H and 35_H

b. $8B_H$ and $D6_H$

c. 79_H and $D2_H$

d. $3C_H$ and 35_H

e. $2E_H$ and $D6_H$

9-17 The CTC is operated in the timer mode with a prescaler factor, PF = 16 and a system clock of 2.5 MHz. The preloaded time constant, TC = 09_H. The elapsed time between the time-out signals is:

a. 5.76 µs
b. 57.6 µs
c. 5.76 ms
d. 576.0 ms

9-18 Refer to question 9-17. If the prescaler factor were 256 instead of 16, the elapsed time between the time-out signals would be:

a. 9.216 µs
b. 92.16 µs
c. 921.6 µs
d. 9216.0 µs

10 Z-80 DIRECT MEMORY ACCESS CHIP

CHAPTER OBJECTIVES

1. To introduce students to the programmable DMA chip.

2. To present the principal features of the three classes of operation in the Z-80/ DMA, namely,

   a. Transfer only,
   b. Search only, and
   c. The combined search-transfer operation.

3. To introduce the modes of operation, namely,

   a. Byte-at-a-time mode,
   b. Burst mode,
   c. The continuous mode, and
   d. The transparent mode.

4. To present the generalized architecture of the DMA chip and describe the function of each entity in the block diagram.

5. To present the package pin assignments and functions in convenient tabular form.

6. To examine the use and operation of the DMA in the daisy chain interrupt scheme of the Z-80 system.

7. To discuss the pulse generation scheme used to keep track of the number of bytes transferred in the DMA operation.

8. To present the bytes used for initialization and the bytes used for control purposes.

9. To discuss the concept of the Variable Cycle Length Time and Auto Restart.

10. To present and discuss the timing involved in various DMA operations such as

a. The variable cycle
b. Command/Control byte write
c. Register read
d. Memory-to-I/O transfers
e. Memory-to-Memory transfers
f. I/O-to-Memory transfers
g. I/O-to-I/O transfers
h. Bus request and acceptance
i. Bus release timing

TEXTBOOK REFERENCES

For reviewing material in the main hardcover textbook, relevant to the topics in this chapter, the following chapter(s) and/or sections are suggested:

1. For general discussion of DMA - Chapter 10

2. For transparent mode - Sec. 10-2.1

3. For cycle stealing mode - Sec. 10-2.2

4. For Byte-at-a-time DMA transfers

   a) Using single memory - Sec. 10-3.1
   b) Using multiple memories - Sec. 10-3.2

5. For Continuous Mode single-block transfers - Sec. 10-4

6. For transfer of multiple data blocks - Sec. 10-5

7. For CPU responses to DMA request- Sec. 10-6.2

8. For operation of the DMA interface chip - Sec. 10-7

9. For byte and block count - Sec. 10-7.3

10. For byte count update - Sec. 10-7.7

11. For block count update - Sec. 10-7.8

10-1 INTRODUCTION

The Z-80/DMA chip is a programmable device which provides the capability to transfer blocks of data between two ports, independent of the CPU. Data block transfers can take place either between the main memory and an I/O device, between two memory

devices, or, between two I/O devices. The DMA chip can also search a block of data for a specified byte (bit maskable). During such a search, a simultaneous transfer can also be accomplished, if desired.

Signals for source and destination addresses, control and timing are provided on the chip. The DMA chip is fully compatible with the rest of the Z-80 system with respect to operations involving bus requests and prioritized interrupt-requests of the daisy chain. It is capable of being directly interfaced to the Z-80 system without requiring any additional external logic.

10-2 PRINCIPAL FEATURES OF THE DMA

10-2.1 General and Electrical Features

1. The DMA is a third generation chip which uses the N-channel, depletion-mode, silicon-gate technology.

2. A single, +5 volts $\pm$5%, power supply is required.

3. A single-phase, TTL level, clock is used.

4. The chip operates in a temperature environment of 0°C to +70°C.

5. All inputs and outputs are TTL compatible.

6. A 40-pin DIP package is used.

10-2.2 Operational Features

1. The DMA device can operate in any of the following three classes of operation:

   a. Transfer Only - Data block is transferred from one port to another, one byte at a time.

   b. Search Only - Data blocks are not transferred. Each byte in a data block is read and compared with contents of two registers, one containing the full Match Byte and the other an optional Mask Byte if only certain desired bits are to be compared.

   c. Search-Transfer Combined - Block of data is transferred until a match is found. Then the transfer operation could be suspended and/or an interrupt request generated.

2. During a transfer, addresses for both the read and the write ports are generated.

3. The timing can be programmed to accomodate the speed any port.

4. Both the Block Length and the Address Registers are double buffered, thereby enabling the values of the next operation to be loaded in the buffer without interfering with the values in the in-process operation.

5. Interrupts can be generated on either Match Found, Ready or End-of-Block under program control.

6. Under program control, the DMA channel can be enabled, disabled or reset.

7. The starting addresses that are programmed for each port (i.e. read and write port), for data transfer or search operations, can be automatically incremented or decremented. These addresses can also remain fixed, if so desired.

8. Complete status of the DMA channel can be determined when requested by the CPU.

9. Without interfering with the data transfer operations, or stopping the transfers, the DMA can signal the CPU when a previously-specified number of bytes have been transferred.

10. Any operation may be made to repeat, in its entirety, either automatically (Auto Restart) or on command (Load).

11. The DMA can be operated at rates up to 1.25 megabytes per second for either the Search or the Transfer operations.

12. The DMA system can be operated in one of the following three modes, under program control, in either the search or the transfer class of operations:

    a. Byte-at-a-Time - Upon request, only one byte is transferred and then control of the system busses is returned to the CPU. A new DMA request is made for each each subsequent byte to be transferred.

    b. Burst Mode - Search or transfer operations continue until such time as the port is not ready and indicates this by deactivating the corresponding Ready line.

    c. Continuous Mode - The search or transfer operation continues until the end of the predetermined data block is reached. If the port is not ready for the operation before end-of-block is reached, the DMA operation pauses until the port is ready and indicates this by reactivating the Ready line.

10-3 THE DMA ARCHITECTURE

1. The generalized block diagram of the DMA chip as shown in Fig. 10-1.

2. The two ports are labelled Port A and Port B and each port can be addressed individually. The address counter of each port is double buffered. The starting address

of the data block to be searched, or transferred, is first loaded into the port Start Address Register of the appropriate port. It is then transferred into its respective counter.

3. Each Port Counter then provides the required address to either the memory or the I/O device on the 16-bit Address Bus (A0 - A15), via a multiplexer, for the byte to be transferred or searched. Under program control the Port Counters can be either incremented or decremented, or their contents can remain unaltered.

4. For data transfer operations, a 16-bit starting address is involved for both ports. The starting addresses involving memories require 2-byte addresses. The starting addresses for I/O ports generally require only one byte and so only the lower byte is used. The 8-bit address for the I/O ports, contained in the Address Register, is usually a fixed address.

5. The 8-bit Status Register contains flags which indicate the current status of the DMA operation. This is a read-only register which indicates status or conditions such as End-of-Block, Byte-Match, Interrupt pending, Ready Active and Write Address Bits valid.

6. The Bus Control and Bus Priority Logic is concerned with three signals which are involved with requests for control of the CPU address bus, the Data Bus and the Status and Control Bus (System Control Bus). This logic also indicates the condition when the system busses have been released for DMA operations as well as conformance to and with the daisy chain priority system.

7. The Interrupt Vector Register holds the 8-bit vector assigned to that particular chip and which is preloaded into this register during the initialization of the chip. During the interrupt acknowledge cycle, the contents of this register are outputted on to the data bus and sent to the CPU provided that the DMA chip is the highest priority device in the daisy chain requesting the interrupt. The vector is then utilized by the CPU for calling up the appropriate service sub-routine.

8. The Interrupt Control and Priority contains logic which is involved in establishing and/or determining the priority of the device in the daisy chain and if it is the highest priority device, then it blocks the lower priority devices from interrupting during execution of the service subroutine.

9. The Control Logic contains control registers which are loaded with specific control bytes for performing the various DMA functions. They are loaded during the chip initialization process. The information contained in these registers include such items as the mode and class of operations and when to initiate an interrupt or other pulse.

10. The Pulse Interval and Pulse Generation Logic contains an 8-bit Pulse Control Register. During initialization, this register is loaded with information on the length of the data block (a data block is defined as a group of data words which

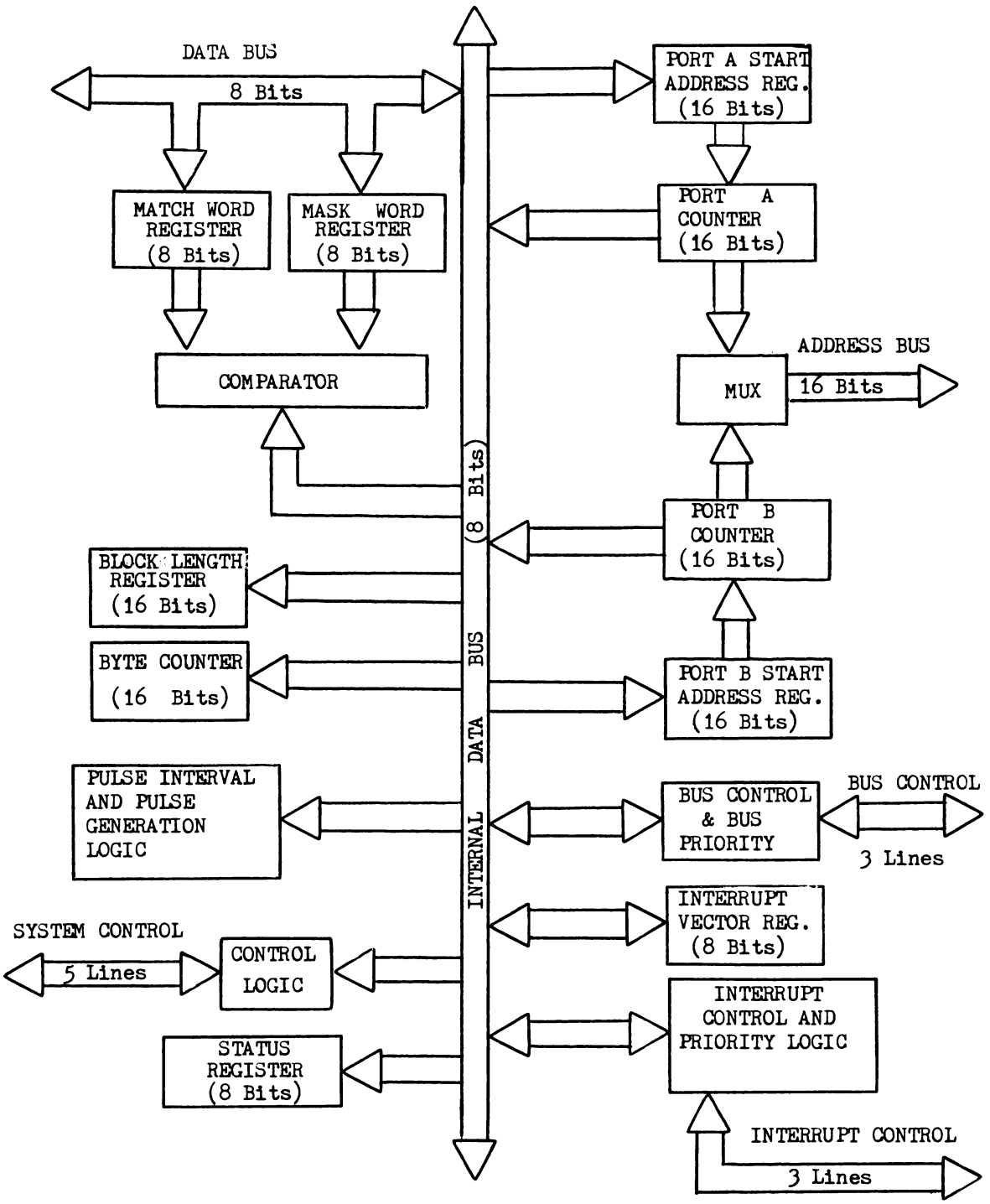


Fig. 10-1 Generalized block diagram of the DMA chip.

is considered as a unit and which are stored in consecutive memory locations) expressed in the number of bytes. When the entire data block is processed, the DMA emits a signal pulse on the interrupt line. This signal is not interpreted by the CPU as an interrupt signal but instead is used to communicate with a peripheral device.

11. The Byte Counter is a 16-bit counter which is cleared at the start of the DMA operation. Then, when the processing of each byte (i.e. either a transfer or a search operation) is completed, the counter is incremented. When the count in the counter matches the contents of the Block Length Register, the End-of-Block flag in the Status Register is set. At this point, depending upon how the DMA has been preprogrammed, the operation may be suspended and/or an interrupt signal generated.

12. The Block Length Register is a 16-bit register which is preloaded during initialization with a number indicating the length (in bytes) of the data block to be processed during the DMA operation, i.e. either the transfer or the search operation.

13. The Match Word Register is an 8-bit register which contains the byte for which a match is to be found during the search operation. When a match is obtained, the Byte-Match flag is set in the Status Register. Note that in this case a bit-by-bit match must be obtained for all 8 bits for the flag in the Status Register to be set.

14. In some cases it is desired that a match be obtained for not all the 8 bits in the match byte but for only certain specific bits in the word. In other words, some bits of the match word are required to be masked out. The word containing the bits to be masked is preloaded into the 8-bit Mask Word Register. Then during the comparison process, only the unmasked bits are examined and compared for a match.

15. The Comparator compares each byte to be processed (i.e. searched or transferred) with the contents of the match Word Register, taking into consideration the bits to be masked as indicated by the Mask Word Register.

10-4 PACKAGE PIN ASSIGNMENTS AND FUNCTIONS

The DMA chip is contained in a standard 40-pin DIP package. The various pins and their respective functions, along with their designations, are presented in convenient tabular form. Functionally, the pins can be divided into seven groups as shown in Fig. 10-2. Note that some lines are shown as unidirectional and others as bidirectional. In the latter group, some of the lines may indicate signal flow in one direction and others in the other direction. The function tables clarify these with the following arrow signals.

DMA ⟶ indicates signals flowing from the DMA.
DMA ⟵ Indicates signals flowing into the DMA.
DMA ⟷ Indicates bidirectional flow on the same line(s).

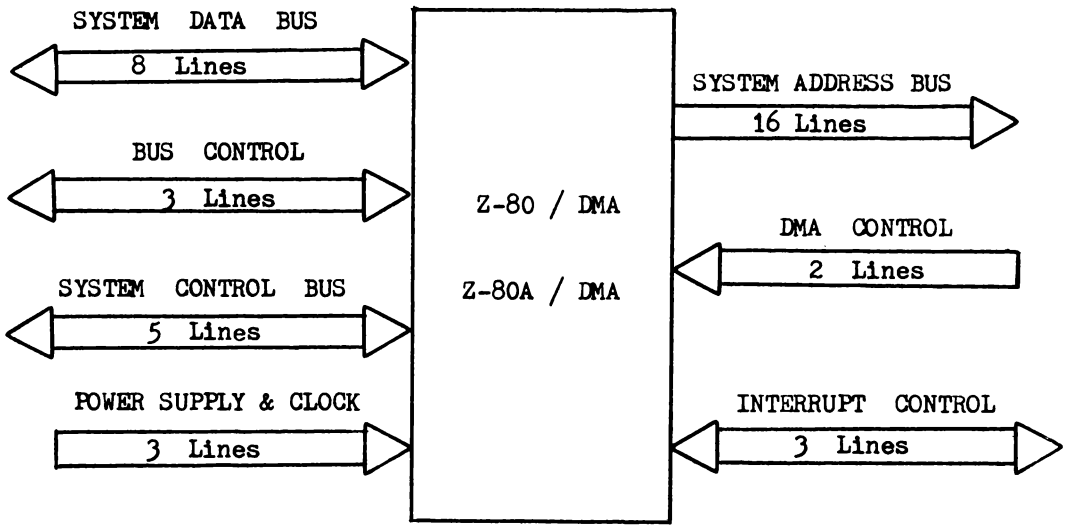


Fig. 10-2 Diagram of DMA package pin groups.

Table 10-1 Data transfer bus group.

PIN DESIGNATIONS	SIGNAL FLOW DIRECTIONS	FUNCTIONS
D0 - D7	DMA ⟷	8-bit, tristate, system data bus, is active high, used for transferring command words from the CPU, DMA status to the CPU and data transfer to and from memories and/or peripheral devices.

Table 10-2 Power Supply and Clock group.

PIN DESIGNATIONS	SIGNAL FLOW DIRECTIONS	FUNCTIONS
ø	DMA ◄——	This is the single-phase, TTL level clock input.
+5V	DMA ◄——	This is the single power supply of +5 ±5% volts that is needed for DMA.
GND	DMA ◄——	This is the power supply ground.

Table 10-3 System address bus group.

PIN DESIGNATIONS	SIGNAL FLOW DIRECTIONS	FUNCTIONS
A0 - A15	DMA ——►	16-bit, tristate, system address bus, is active high. Provides direct addressing capability for 64K byte memory and upto 256 I/O ports for DMA, transfer and search of data blocks.

Table 10-4 Interrupt control group.

PIN DESIGNATIONS	SIGNAL FLOW DIRECTIONS	FUNCTIONS
$\overline{INT}$/PULSE	DMA ⟶	Interrupt Request signal, active low. The DMA requests interrupt service from the CPU.
IEI	DMA ⟵	Interrupt Enable In, active high, used in the priority interrupt daisy chain. A high or active signal indicates that no other I/O device with higher priority is currently being serviced.
IEO	DMA ⟶	Interrupt Enable Out, active high. Also used in the priority interrupt daisy chain. It is active only when IEI is active and the CPU is not servicing an interrupt from the DMA. When active, it blocks interrupt signals from lower priority I/O devices from interrupting while it is being serviced.

Table 10-5 Bus control group.

PIN DESIGNATIONS	SIGNAL FLOW DIRECTIONS	FUNCTIONS
$\overline{\text{BUSRQ}}$	DMA ◄——►	Bus request, active low. As output signal it requests the CPU to relinquish control of the system data bus, the system address bus and the control bus. When more than one DMA chip is used in the daisy chain, this signal is an input signal which prevents this DMA from making bus requests to CPU until the DMA operation of the other DMA is completed.
$\overline{\text{BAI}}$	DMA ◄——	Bus Acknowledge In, active low. It is an input from the CPU informing the DMA chip that the CPU has released the busses. If more than one DMA chip is connected on the daisy chain, then the acknowledge pin of the CPU is connected to the $\overline{\text{BAI}}$ pin of only the highest priority DMA chip. The $\overline{\text{BAI}}$ pins of the lower priority DMA chips are connected to the $\overline{\text{BAO}}$ pin of the previous higher priority DMA chip.
$\overline{\text{BAO}}$	DMA ——►	Bus Acknowledge Out, active low, and is used when more than one DMA chip is connected in the daisy chain. When active, it signals that no other higher priority DMA chip has currently requested control of the system busses. $\overline{\text{BAO}}$ pin receives its input from the $\overline{\text{BAI}}$ pin of the next higher priority DMA chip in the daisy chain.

Table 10-6 System control bus group.

PIN DESIGNATIONS	SIGNAL FLOW DIRECTIONS	FUNCTIONS
$\overline{M1}$	DMA ◄———	Machine Cycle One signal, active low. When $\overline{M1}$ and $\overline{RD}$ are simultaneously active, the CPU is fetching an instruction from the memory. When M1 and $\overline{IORQ}$ are active simultaneously, the CPU is scknowledging an interrupt. When $\overline{M1}$ is active without the other two being active, it is used by the DMA to decode the 2-byte return-from-interrupt (RETI) instruction, whose HEX codes are ED and 4D.
$\overline{IORQ}$	DMA ◄——►	Tristate, I/O request signal is active low. When $\overline{IORQ}$ and $\overline{M1}$ are simultaneously active, the CPU is acknowledging an interrupt. When active by itself as an input, it informs the DMA that the lower 8 bits of the address bus are currently holding a valid I/O port address and that transfer of either the status word from the I/O port to the CPU or the transfer of a control byte from the CPU to the port may take place. Also as an input, it is simultaneously active with $\overline{CE}$, $\overline{RD}$ and $\overline{WR}$ signals, it informs the DMA that this particular chip is the addressed port. After the DMA chip has acquired control of the system busses, it outputs a signal on the $\overline{IORQ}$ pin indicating that the lower 8 bits of the address bus contain a valid address for the I/O device which is involved in the transfer of data.
$\overline{MREQ}$	DMA ◄——►	Memory Request, active low, tristate signal. For a memory read or write operation, it indicates that a valid memory address is available on the system address bus. After the DMA chip has obtained control of the system busses, this signal requests a data transfer either to or from the memory.
$\overline{RD}$	DMA ◄——►	Tristate, active low, Read signal. An input from the CPU on this pin results in a readout of the status words from the DMA register. The DMA outputs a signal on this pin, after it has obtained control of the system busses, to request a read out from either the addressed memory location or the addressed I/O port.
$\overline{WR}$	DMA ◄——►	Tristate, active low, Write signal. An input signal on this pin from the CPU informs the DMA that the CPU wants to write control or command words into the appropriate registers of the DMA chip. After the DMA chip has obtained control of the system busses, it outputs a signal on this pin to indicate a DMA write operation on either the addressed memory location or the addressed I/O port.

Table 10-7 DMA control group.

PIN DESIGNATIONS	SIGNAL FLOW DIRECTIONS	FUNCTIONS
RDY	DMA ◄———	Ready signal input which, under program control can be made either active high or active low. The DMA chip monitors this input to establish whether a peripheral connected to a DMA port is ready for a transfer operation (read or write). Also depending upon the mode of DMA operation, this signal indirectly controls the DMA activity by activating the $\overline{\text{BUSRQ}}$ line, i.e. go either low or high.
$\overline{\text{CE/WAIT}}$	DMA ◄———	Chip enable, active low, signal from the CPU. It is active when $\overline{\text{WR}}$ and $\overline{\text{IORQ}}$ are simultaneously active. The address on the system address bus is the address of the DMA. This allows a command or control word to be transferred from the CPU to the DMA chip. The same pin can be programmed as a $\overline{\text{WAIT}}$ signal input from the memory or the peripheral device. As a $\overline{\text{WAIT}}$ signal. it inserts wait states in the DMA cycles after the DMA has received an acknowledge from the CPU in response to its Bus request. The insertion of the wait states slows the DMA and establishes speed compatibility with slower memory and I/O devices.

Table 10-8 Pin number assignments.

PIN DESIGNATIONS	PIN NUMBERS	PIN DESIGNATIONS	PIN NUMBERS
System Address Bus		System Data Bus	
A0	6	D0	35
A1	5	D1	34
A2	4	D2	33
A3	3	D3	32
A4	2	D4	31
A5	1	D5	29
A6	40	D6	28
A7	39	D7	27
A8	24		
A9	23	System Control Bus	
A10	22		
A11	21	$\overline{M1}$	26
A12	20	$\overline{IORQ}$	10
A13	19	$\overline{MREQ}$	12
A14	18	$\overline{RD}$	9
A15	17	$\overline{WR}$	8
Bus Control		Interrupt Control	
$\overline{BUSRQ}$	15	$\overline{INT}$/PULSE	37
$\overline{BAI}$	14	IEI	38
$\overline{BAO}$	13	IEO	36
Power Supply & Clock		DMA Control	
ø	7	RDY	25
+5V	11	$\overline{CE/WAIT}$	16
GND	30		

10-5 FUNCTIONAL OPERATION OF THE DMA

The Z-80/DMA is capable of performing three different functions referred to as the three classes of operations. Additionally, it can be programmed to operate in three different modes. The classes and modes of operations are briefly described in this section.

10-5.1 The Classes of Operations

Basically, the DMA can be programmed to perform any of the three functional operations which involve transfer of data between two ports and/or search for a specific byte (or certain bits of a byte). The three classes of operations can be explained by referring to Fig. 10-3 below.

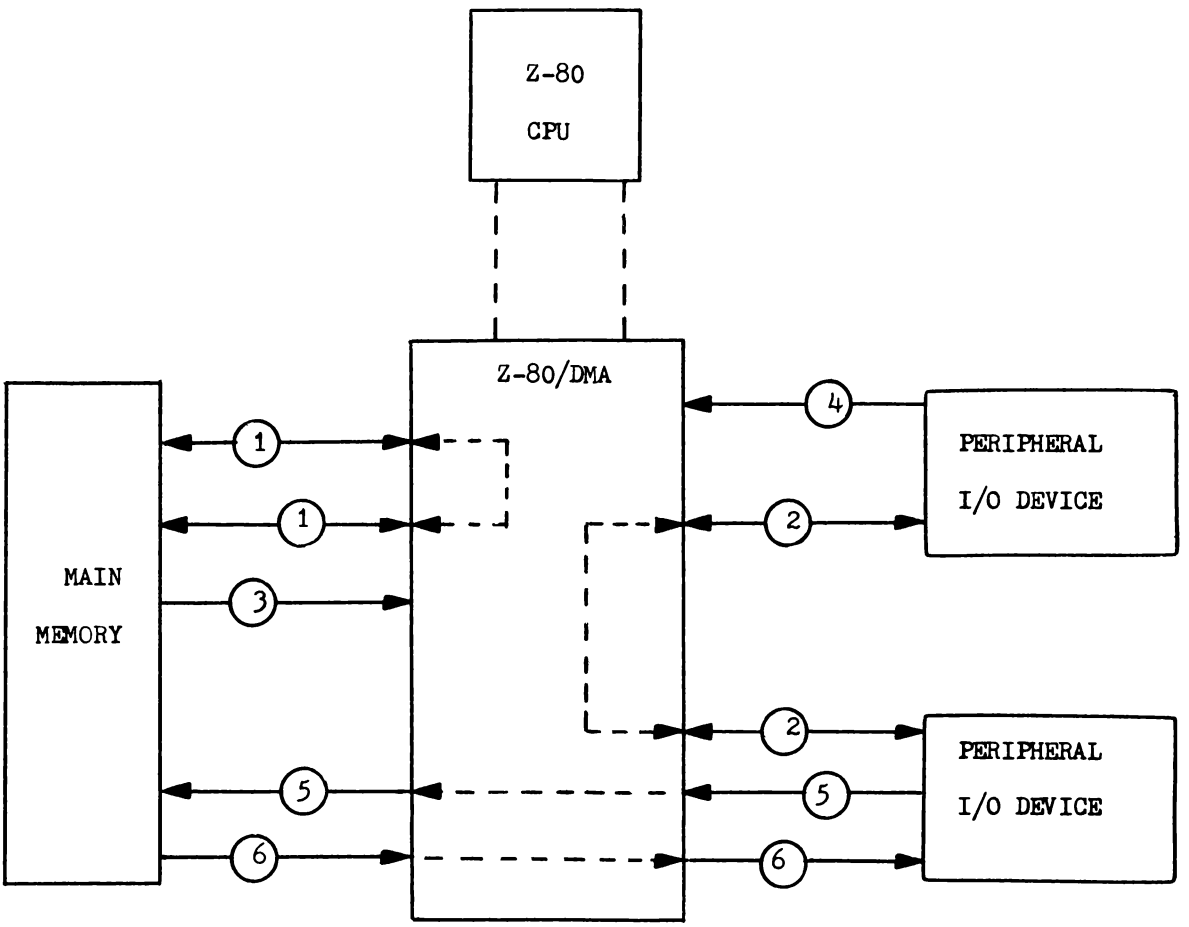


Fig. 10-3 The three classes of operations.

1. Data Transfers between Ports - Blocks of data can be transferred from a designated source port to a designated destination port. In Fig. 10-3, the transfers indicated are as follows:

   a. Memory-to-memory transfers (lines 1)
   b. I/O-to-I/O transfers (lines 2)
   c. I/O-to-memory (lines 5)
   d. Memory-to-I/O transfers (lines 6)

   In the DMA chip the two ports are labelled Port A, and Port B. Both ports are addressable and data are transferred from the source to the destination port, one byte at a time. The memory-to-memory transfer capability makes it possible to relocate blocks of data from one part of the main memory to another.

2. Search Operations - Blocks of data, located either in the memory (line 3) or in the peripheral device (line 4), can be called out, one byte at a time, and compared with the contents of two previously loaded internal registers in the DMA chip. The first register contains a Match byte and the second register contains a Mask Byte which, if desired, can mask out certain bits in the Match byte. This enables the user to conduct a block search only on certain desired bits instead of the whole byte. Using the 2.5 MHz Z-80/DMA, search rates of up to 1.25 M bytes per second can be obtained. With the 4.0 MHz Z-80A/DMA, search rates up to 2 M bytes per second are obtained.

3. Combined Transfer-Search Operations - This class of operation combines the transfer and search operations, described in paragraphs 1 and 2 above. A block of data is transferred from one port to another as mentioned in paragraph 1. As each byte is transferred, it is simultaneously compared with the Match byte (which may contain selectively masked bits, if so desired). When a match is obtained, the transfer is suspended and/or an interrupt generated depending upon how the DMA chip is programmed.

10-5.2 Modes of Operation

Under program control the DMA can be made to operate in one of the following four modes.

1. Byte-at-a-Time Mode - In this mode, the DMA operation is performed on one byte of the data block at a time. Upon completion of the DMA operation on a single byte, the control of system busses is returned to the CPU. For each of the succeeding DMA operations, a new request for the busses is made.

2. Continuous Mode - In this mode, once the DMA operation is started it continues the processing of the entire programmed data block. If the particular port's READY line goes inactive, before the end of the block is reached, the DMA does not release the system busses but just pauses until the READY line goes into the active state again.

3. Burst Mode - In this mode, the DMA operation continues as long as the port's READY line is in the active status. If it goes into the inactive status before the end of the programmed data block, the DMA operation stops and control of the system busses is returned to the CPU after completing the operation on the current byte.

4. Transparent Mode - In this mode, the DMA operation takes place during the time frames when the memory refresh operation takes place. Effectively, the DMA operations are interleaved with CPU operations and consequently the CPU does not lose any time because of the DMA operations.

5. Block Lengths - In the Z-80/DMA, high-speed buffering is used to read the data bytes from a port. The design of the data read system is such that the DMA operations on the first byte in are not completed until the following byte is read in. This has the following two implications. First, the data block upon which a DMA operation is to be performed must be two or more bytes long. Second, the block length loaded into the 16-bit Block Length Register (See Fig. 10-1) must be one less than the actual length of the block.

Note: In all the various modes, the DMA operation on the current byte (i.e. the byte that is read in) is always completed regardless of the status of the port's READY line or the status of other control signals involved in the operation.

10-5.3 The Interrupt System

The DMA chip is also used in the daisy chain interrupt system of the Z-80. Under program control, the DMA can interrupt the CPU under any of the following conditions:

1. Generate an interrupt on Ready, before requesting control of the busses,

2. Generate an interrupt when a Match is obtained when operating in the Search Mode or combined Transfer-Search Mode.

3. Generate and interrupt when the end of the data block is reached.

Note that because of the peculiarity in buffering the incoming data bytes (discussed in paragraph 5 of Section 10-5.2), an interrupt generated by a Match at the end of the data block will be generated if a match is obtained on the byte just prior to the very last byte of the block. The interrupt then is generated just before the last byte in the block.

Another feature of the Z-80/DMA interrupt system is that when the interrupt is generated, it sets an interrupt pending bit which can, under program control, alter the interrupt vector assigned to that DMA chip.

10-5.4 Pulse Generation

In some applications it is desirable for the peripheral I/O device to keep track of how many bytes have been transferred in a DMA operation. In the Z-80/DMA, a pulse is outputted at every 256-byte transfer interval. This signal is outputted on the interrupt line but is not interpreted as an interrupt by the CPU because it appears when both the Bus Request ($\overline{BUSRQ}$) and the Bus Acknowledge ($\overline{BAI}$ or $\overline{BAO}$) lines are active. (See Table 10-5.)

10-5.5 Command and Control Bytes

The DMA chip contains several registers into which specific words can be loaded or written. Bytes can be loaded into these registers only when the DMA is not controlling the system busses. During the process of loading the write registers, the DMA is disabled and remains in that condition until it is enabled by means of a specific command. These registers fall into the following two categories:

1. Initialization Bytes - These bytes are used for initially setting up the DMA chip. They include the following commands:

   Class of operation
   Mode of operation
   Starting address of data block
   Length of data block
   The Match byte
   The Mask byte
   Interrupt Vector
   Status conditions affecting interrupt vectors
   Pulse counting for bytes transferred
   Interrupt conditions
   Rules for Read-Line and Wait-Line

2. Control Bytes - These bytes are loaded into write registers and they initiate immediate actions by the DMA chip. They include the following:

   Load Starting Address buffers
   Clear counters
   Clear status bits
   Enable
   Disable
   Reset

10-5.6 Reading from Internal Registers

1. The Z-80/DMA has seven internal registers whose contents can be read out. The contents of these registers can be read out in the following sequence:

a. The 8-bit Status Register
b. Block Length Register, low byte
c. Block Length Register, high byte
d. Port A Address Register, low byte
e. Port A Address Register, high byte
f. Port B Address Register, low byte
g. Port B Address Register, high byte

2. The sequential read out of these 7 registers is accomplished by means of an internal pointer which points to each register successively after each READ operation.

3. It is possible to skip the read-out of any desired register by inserting a 0 in the appropriate bit position in the READ byte.

10-5.7 Variable Cycle Length Time

To accomodate the different operating speeds of various memories and peripherals used in the DMA operations, the Z-80/DMA chip has unique features which allows the user to vary the cycle time length. The cycle length can be varied from one to four T-cycles. (Recall that a T-cycle is defined as one clock period). For situations where additional time is desired, additional $\overline{\text{WAIT}}$ states can be inserted. This feature has two principal advantages. First, it allows the designer to adapt the DMA operating speeds compatible with the speed of the various other system components. Second, it eliminates the need for additional external logic for conditioning the associated control signals. The various timing relationships are shown later in this chapter.

10-5.8 Port Addressing

For every DMA transfer, two 16-bit addresses are generated by the DMA chip; one address is for the source port and the other for the destination port. Each address can be either fixed or variable. The variable address scheme allows the port address to be sequentially incremented or decremented from a starting address. The length of the block of data to be transferred, expressed in number of bytes, is loaded into the Block Length Register. Block length of up to 64K bytes can be used. During a DMA transfer, a port address is first generated for the source port and then another for the destination port. These addresses are time-multiplexed onto the system address bus.

10-5.9 Auto Restart

This feature automatically reloads the starting address for either of the two ports upon completion of a block transfer. When the addresses are reloaded, the byte counter is cleared. This feature is activated by an Auto Restart Control Bit in one of the command words.

This feature reduces the software burden on the CPU in applications which require repetitive operations. Also, if Byte-at-a-Time or the Burst Mode is selected for data transfers, the CPU has access to the system busses for certain periods of time. During such time periods, it is possible to write new or different block starting addresses in the DMA chip. Thus, an Auto Restart will start a block transfer or search operation at a new starting address.

10-6 THE DMA TIMING

10-6.1 The Variable Cycle Timing

Previously, in Section 10-5.7, we had briefly described the variable cycle timing of the Z-80/DMA. In addition to allowing the user to vary the cycle length time, a unique feature allows the four control signals associated with each port (Namely $\overline{IORQ}$, $\overline{MREQ}$, $\overline{RD}$ and $\overline{WR}$) to have their trailing, positive-going, edges terminated one-half T-cycle early if so desired.

In Fig. 10-4, the variable timing waveforms are shown as programmed for four T-cycles. The earlier termination of the four control signals is shown in dotted lines for each signal. This feature gives the user an added dimension of flexibility. The shorter duration of the read and write control signals allows them to be terminated before the transferred data begins to change. Also shown in Fig. 10-4 are the points at which $\overline{WAIT}$ signals from either the memory or the I/O device will come in during the lengthened timing cycle.

10-6.2 DMA Inactive State Timing

10-6.2.1 Command/Control Byte Write Cycle

Figure 10-5 shows the timing waveforms associated with the process of writing the command or control byte into one of the registers in the DMA chip. This is a CPU-to-DMA transfer of one byte which involves activation of three control signals, namely $\overline{CE}$, $\overline{IORQ}$ and $\overline{WR}$ which are activated simultaneously. Because of this and to simplify the timing waveforms, only one waveform is shown in Fig. 10-5 for all three control signals. The command or control byte is then outputted by the Z-80/CPU approximately one T-cycle later. Note that during this operation, the DMA is in an inactive or disabled state and so the CPU addresses it and writes into it as if it were an I/O device.

10-6.2.2 DMA Register Read Cycle

The timing waveforms shown in Fig. 10-6 are for the operations DMA-to-CPU read-outs from the DMA Status Register, the Port Address and the Byte Counters. The three control signals involved in this operation are $\overline{CE}$, $\overline{IORQ}$ and $\overline{RD}$, and since they are activated simultaneously, they are shown as only one waveform in Fig. 10-6 for simplicity.

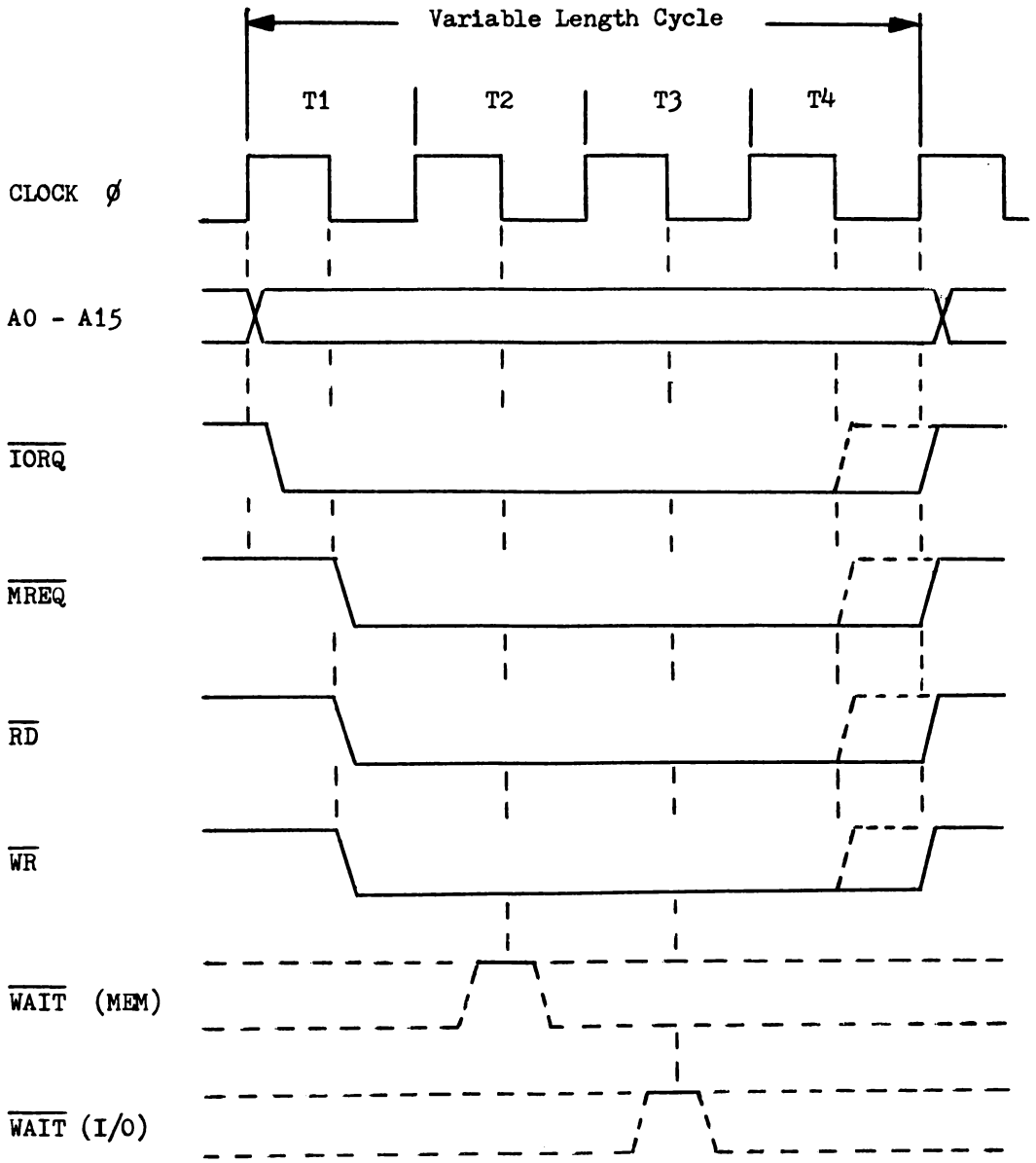


Fig. 10-4 Variable cycle length timing. (programmed for 4 T-cycles)

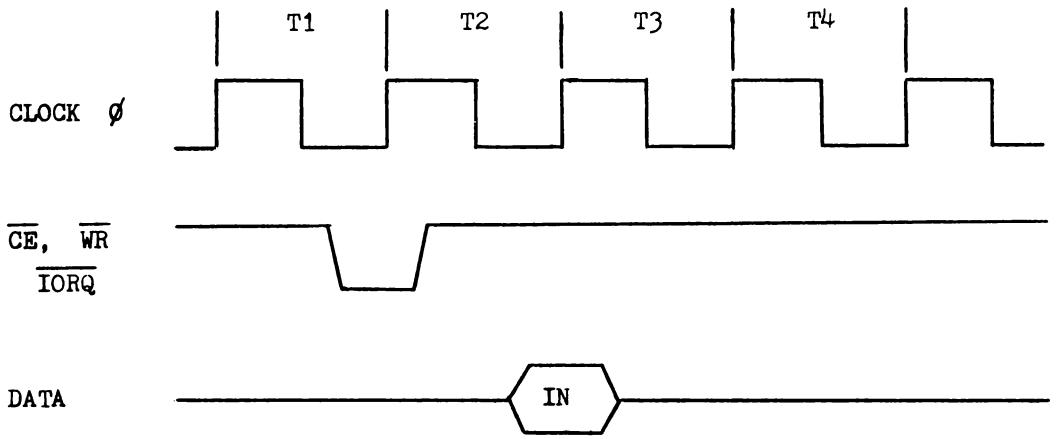


Fig. 10-5 CPU-to-DMA command byte write timing.

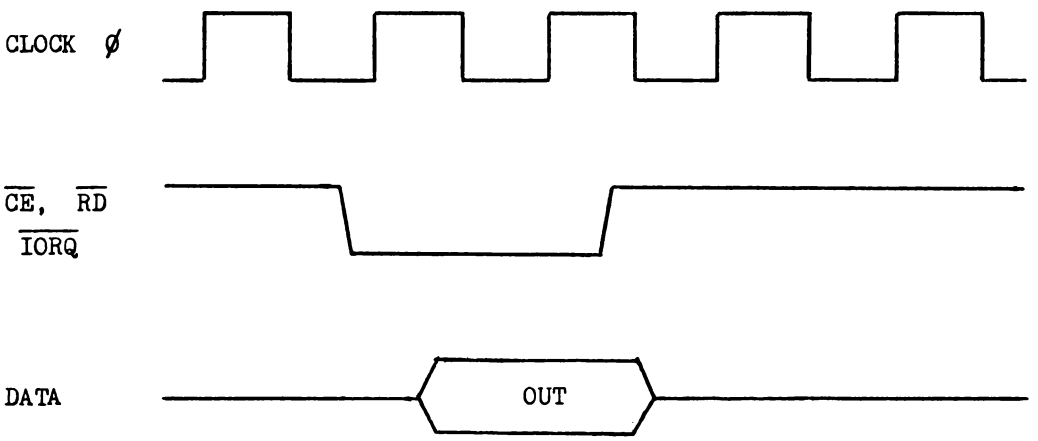


Fig. 10-6 Read-out timing. (DMA-to-CPU)

The data appears on the bus approximately one T cycle after the control signals are activated.

10-6.3 DMA Active State Timing

10-6.3.1 Memory-to-I/O Transfers

The memory-to-I/O transfers involve reading data bytes from the memory port,

transferring them internally to the other port and then writing them into the I/O device or peripheral. After resetting the DMA chip, the read and write timing in the DMA operation are identical to the CPU read and write timings for the memory and the I/O devices. There is one exception, clearly shown in Fig. 10-7, which is the timing waveform chart for the memory-to-I/O data transfers. During the memory read cycle, the data bytes are latched on to the D0 - D7 bus during the negative-going trailing edge of T3 of this cycle. These bytes are then retained on the system data bus until the end of the I/O write cycle, i.e., until after completion of the following T3 cycle.

Notice that the memory read cycle consists of only three T-cycles whereas an extra WAIT state is inserted between T2 and T3 of the I/O write cycle. This is done primarily to reconcile the timing differences between faster memories and slower I/O devices. To further accomodate slower peripherals, the following procedure is used:

1. The $\overline{\text{CE/WAIT}}$ is programmed as a $\overline{\text{WAIT}}$ line when the DMA chip is operating in the active state.

2. The $\overline{\text{CE/WAIT}}$ line is sampled during the negative-going trailing edge of T2 in the memory read cycle and the negative-going trailing edge of TW in the I/O write cycle.

3. If the $\overline{\text{CE/WAIT}}$ line is low when it is sampled, it indicates that the I/O or the memory is not ready for a byte transfer.

4. The DMA then automatically inserts a TW cycle either after T2 of the memory read cycle, or the TW cycle of the I/O write cycle, or both, as appropriate.

5. The $\overline{\text{CE/WAIT}}$ line is then sampled again on the negative-going, trailing edge of the newly-inserted TW cycle. If the CE/WAIT is still low, another TW cycle is inserted. The process is repeated until the I/O or the memory is ready for the transfer and indicates this by a high signal on the $\overline{\text{CE/WAIT}}$.

10-6.3.2 Memory-to-Memory Transfers

The timing involved in memory-to-memory DMA transfers is shown in Fig. 10-8. Notice that both the memory read and the memory write cycles each have three T-cycles. Also, an active $\overline{\text{MREQ}}$ signal is involved in both the read and the write cycles. The $\overline{\text{RD}}$ and the $\overline{\text{WR}}$ signals identify the read and the write cycles respectively. As in the previous case of memory-to-I/O DMA transfers, wait states can be inserted between the T2 and T3 of either read or write cycle, or both, if the $\overline{\text{CE/WAIT}}$ line is low during the middle of T2. If the $\overline{\text{CE/WAIT}}$ line were not high, as shown in Fig. 10-8, then a wait state would have been inserted between T2 and T3.

10-6.3.3 I/O-to-Memory Transfers

Figure 10-9 shows the timing involved in the I/O-to-memory DMA transfers. Notice that a wait state is automatically inserted in the I/O read cycle but not in the memory read cycle. Of course additional wait states can be inserted in either cycle if the $\overline{\text{CE/WAIT}}$ is low at the appropriate time.

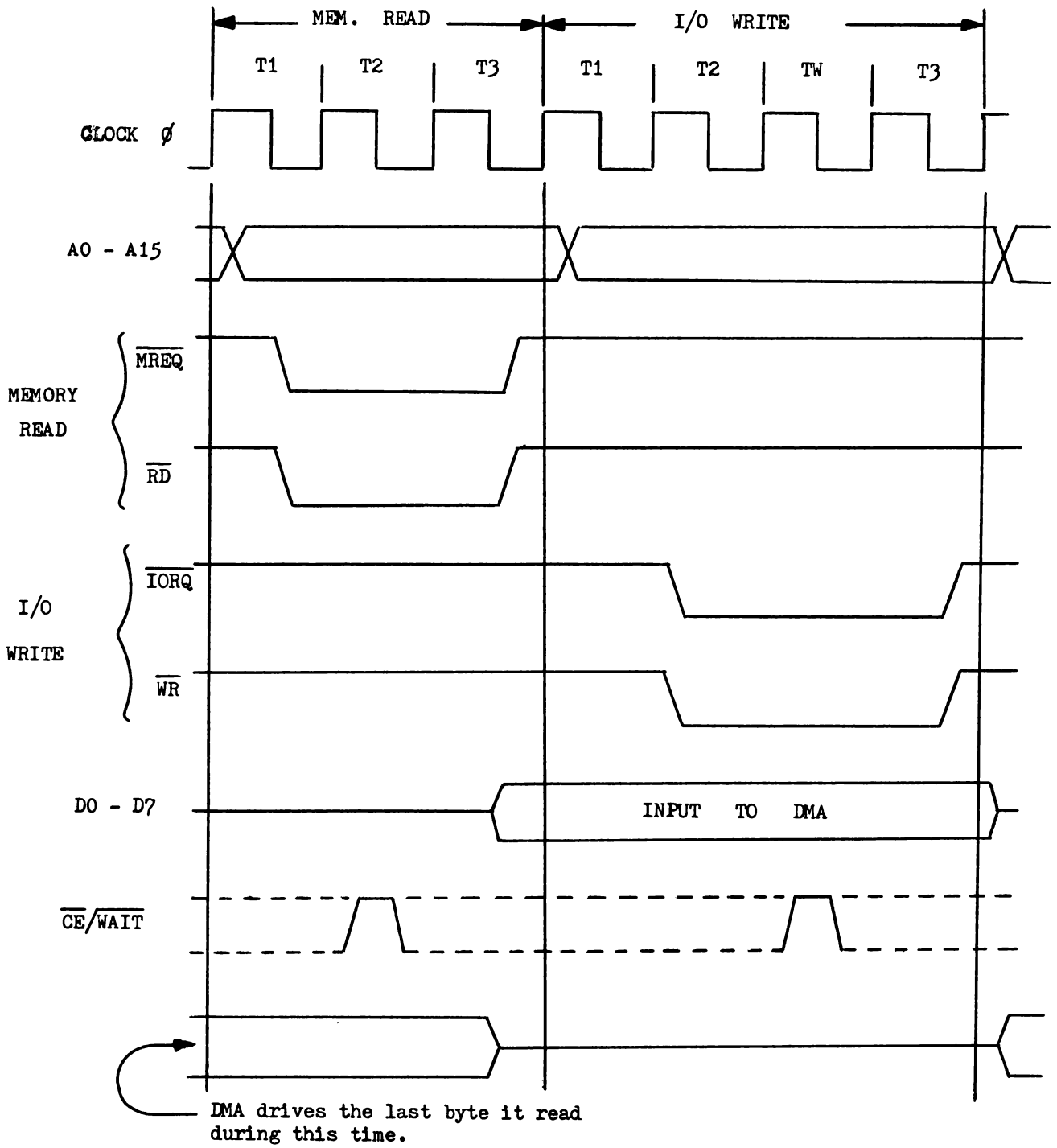


Fig. 10-7 Memory-to-I/O transfer timing.

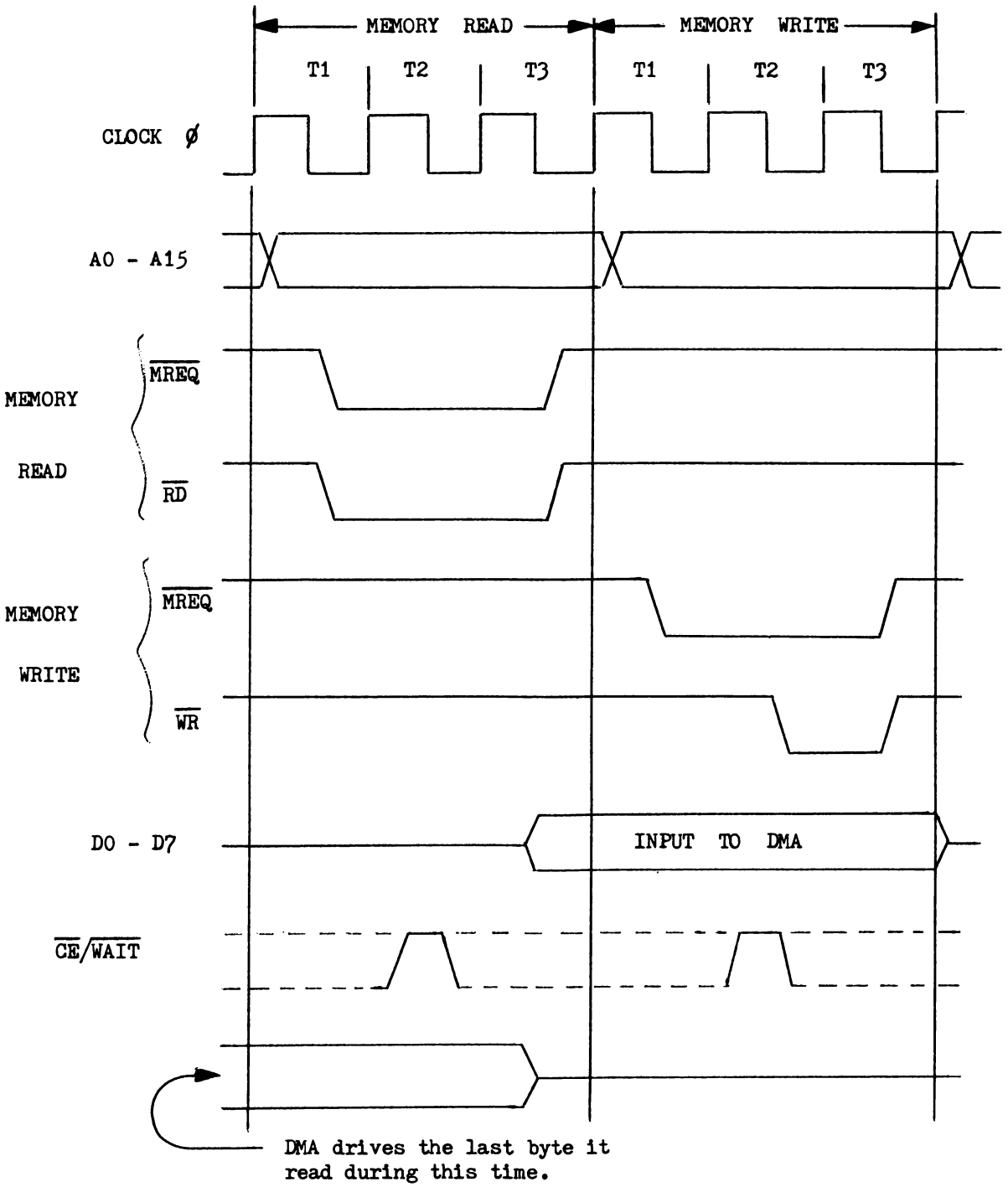


Fig. 10-8 Memory-to-Memory transfer timing.

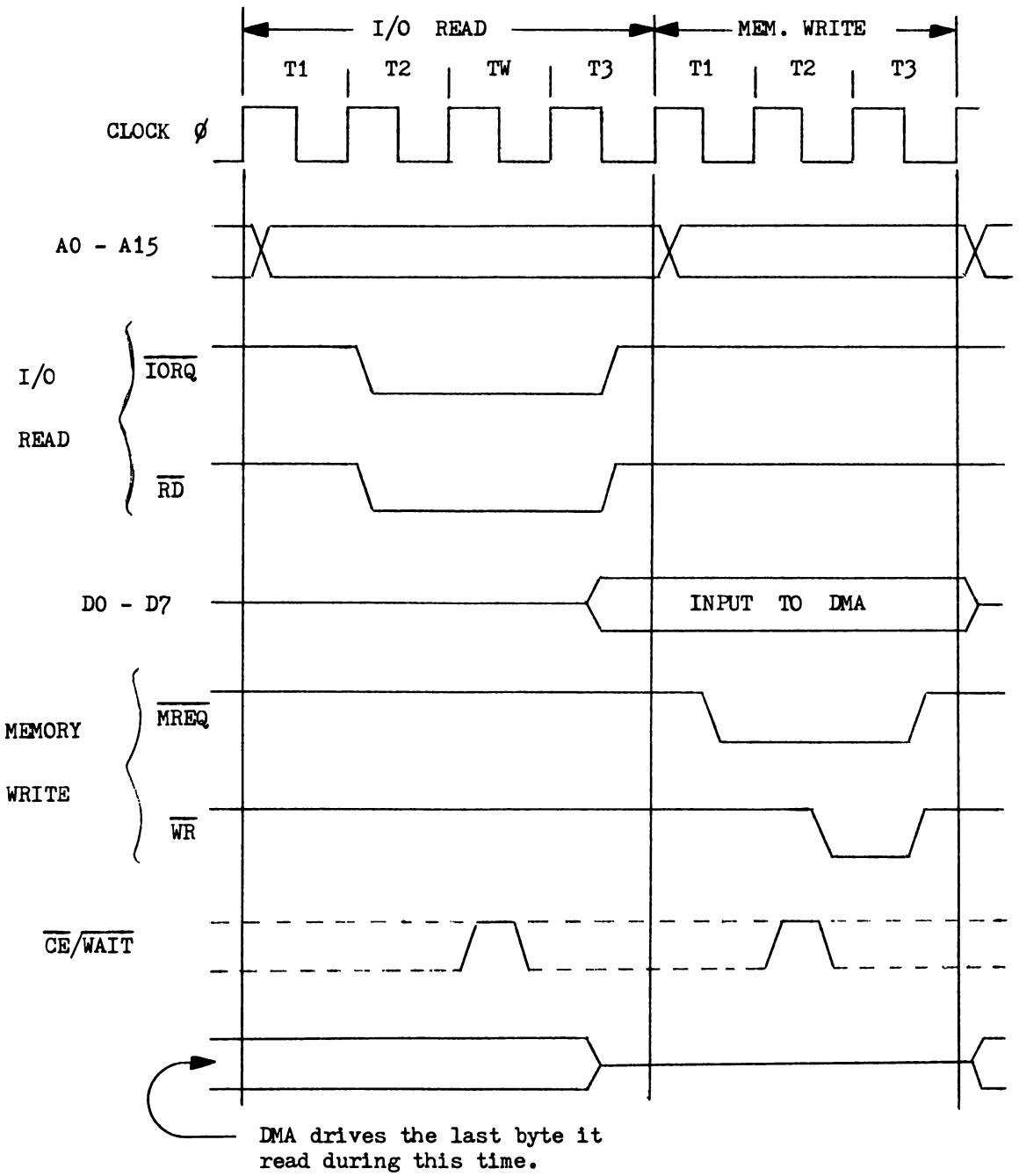


Fig. 10-9 I/O-to-Memory transfer timing.

10-6.3.4 I/O-to-I/O Transfers

The timing involved in I/O-to-I/O DMA transfers is shown in Fig. 10-10. In this case, both the I/O read and write cycles have one wait state automatically inserted between T2 and T3. Of course, additional wait states can be inserted in either or both cycles when the CE/WAIT line is low.

10-6.4 Bus Request and Acceptance Timing

The DMA Bus Request and Acceptance timing for the Byte-at-a-time, Burst and Continuous modes are shown in Fig. 10-11. The DMA chip samples the ready input line (RDY), which can be programmed active high or low, on the negative-to-positive edge of every clock Ø to determine if the peripheral associated with a DMA port is ready for a transfer. If RDY is active, a $\overline{\text{BUSRQ}}$ signal is generated and sent to the CPU on the rising edge of the next clock Ø. The CPU then responds with a bus acknowledge signal to the $\overline{\text{BAI}}$ line to the DMA either directly or through the daisy chain. The $\overline{\text{BAI}}$ line is sampled on the leading edge of every clock Ø. When a BAI active state is detected for two consecutive positive-going edges of the clock, the DMA goes into the active state on the next leading edge of Ø as in Fig. 10-11.

10-6.5 Bus Release Timing for Burst and Continuous Modes

10-6.5.1 At End of Block

When operating in either the burst or the continuous mode, the $\overline{\text{BUSRQ}}$ line goes high when the end of the data block is reached. Usually, this happens on the same leading edge of the clock Ø at which the DMA completes the transfer of the data block. The DMA then goes into an inactive or disabled state, as shown in Fig. 10-12. Notice that the transfer of the last byte in the data block is completed regardless of whether the RDY is active or inactive during this transfer.

10-6.5.2 On "Not Ready"

When operating in the burst mode, if the RDY signal becomes inactive, the DMA completes its current operation on the in-process byte and then causes the $\overline{\text{BUSRQ}}$ line to go high (i.e. inactive) on the next leading edge of the clock Ø. This is shown in Fig. 10-13.

When operating in the continuous mode, the $\overline{\text{BUSRQ}}$ line is held in the active state or low when the RDY line becomes inactive. In this case, the DMA just idles after completing the current operation until such time when RDY again goes in the active state.

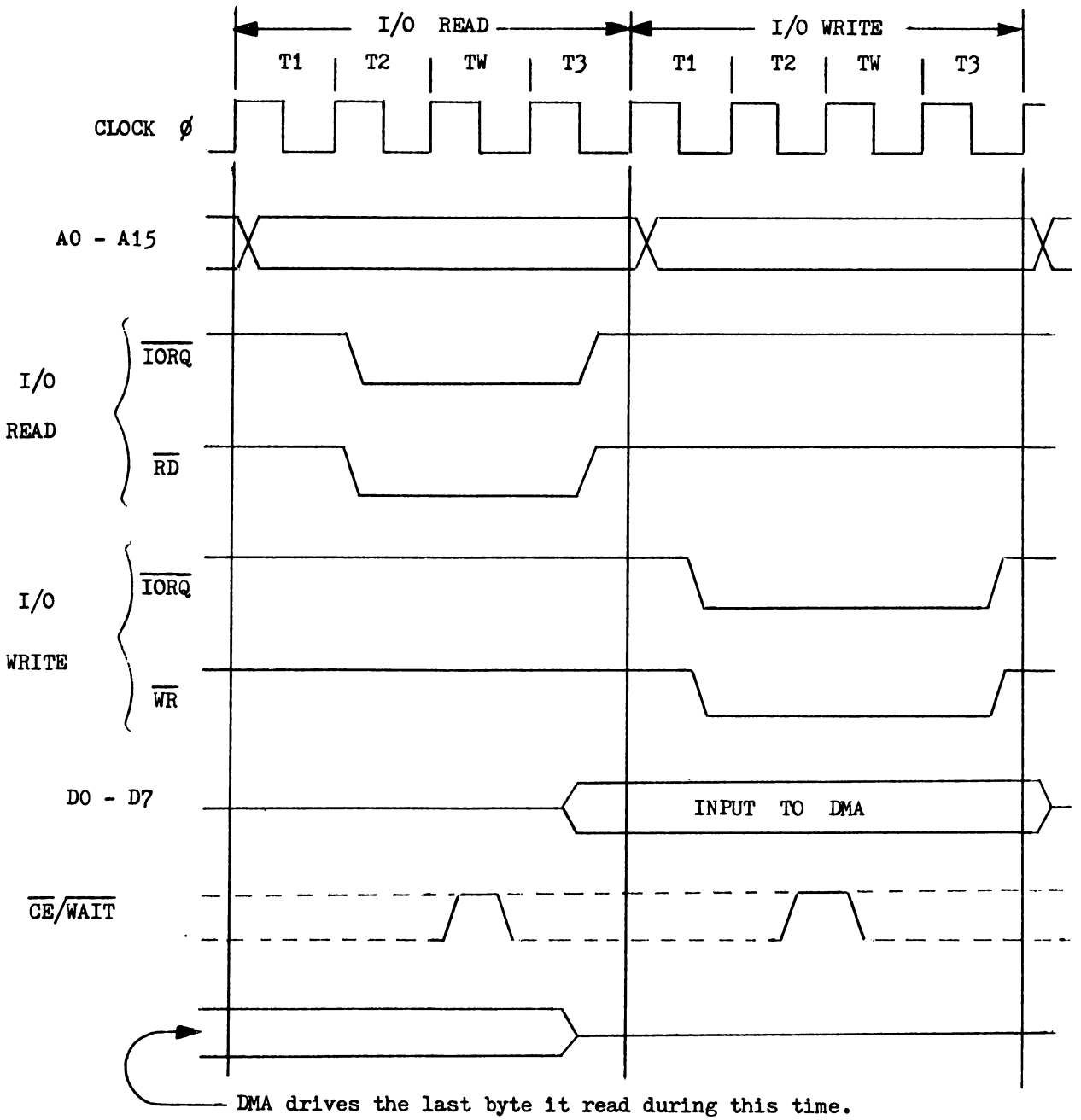


Fig. 10-10 I/O-to-I/O transfer timing.

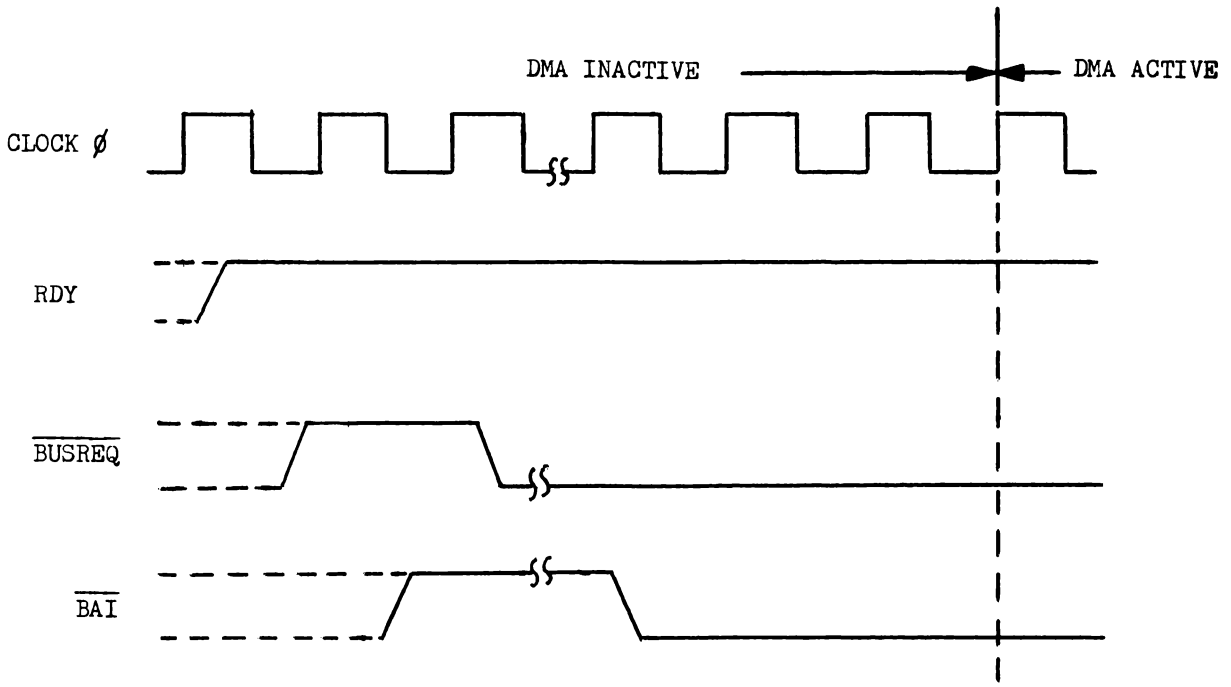


Fig. 10-11 Bus Request and Acceptance for all modes.

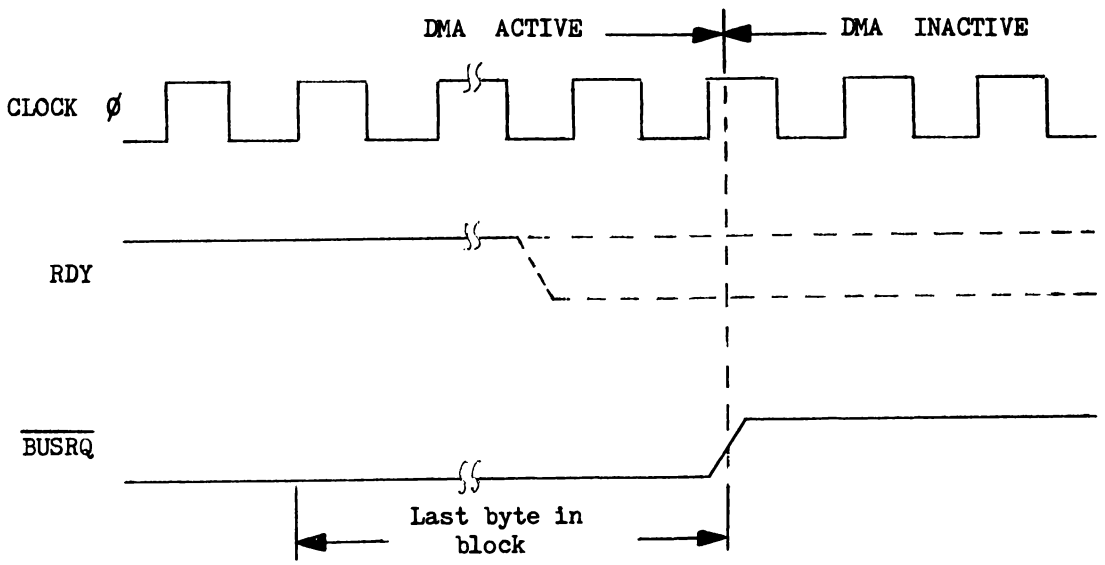


Fig. 10-12 Bus Release at End-of-Block timing for Burst and Continuous Modes.

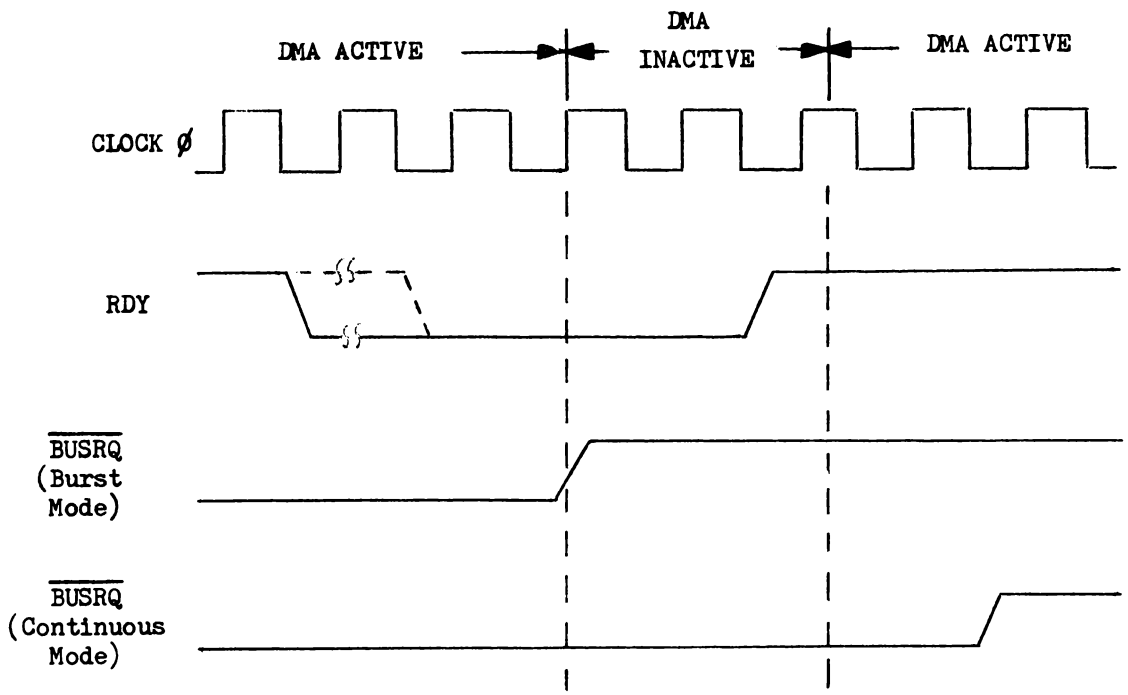


Fig. 10-13 Bus Release timing when "Not Ready" for Burst and Continuous Mode.

10-6.5.3 On "Match"

When operating in either the burst or the continuous mode, the DMA can be programmed to stop when a match is obtained by comparing the data byte to the match byte. When a match is found, the $\overline{\text{BUSRQ}}$ goes high or inactive but only after completing the DMA operation on the next byte which could be either a read in a search process, or a write in a transfer process. Recall that because of the pipelining process (i.e. double buffering) used in the DMA chip, matches are performed while the next DMA read or write operation is being executed. This is shown in Fig. 10-14. Note that the RDY line can become inactive any time after the match operation without affecting bus release.

10-6.6 Bus Release Timing for Byte-at-a-Time Mode

When operating in the Byte-at-a-time mode, the $\overline{\text{BUSRQ}}$ signal goes high on the leading edge of the clock Ø prior to the end of each read cycle if operating in the search-only class. If the DMA is operated in the transfer or the combined search-transfer class, the $\overline{\text{BUSRQ}}$ line goes high on the leading edge of the clock Ø prior to the end of each write cycle. This is shown in Fig. 10-15. Note that $\overline{\text{BUSRQ}}$ goes high regardless of the status of the RDY line.

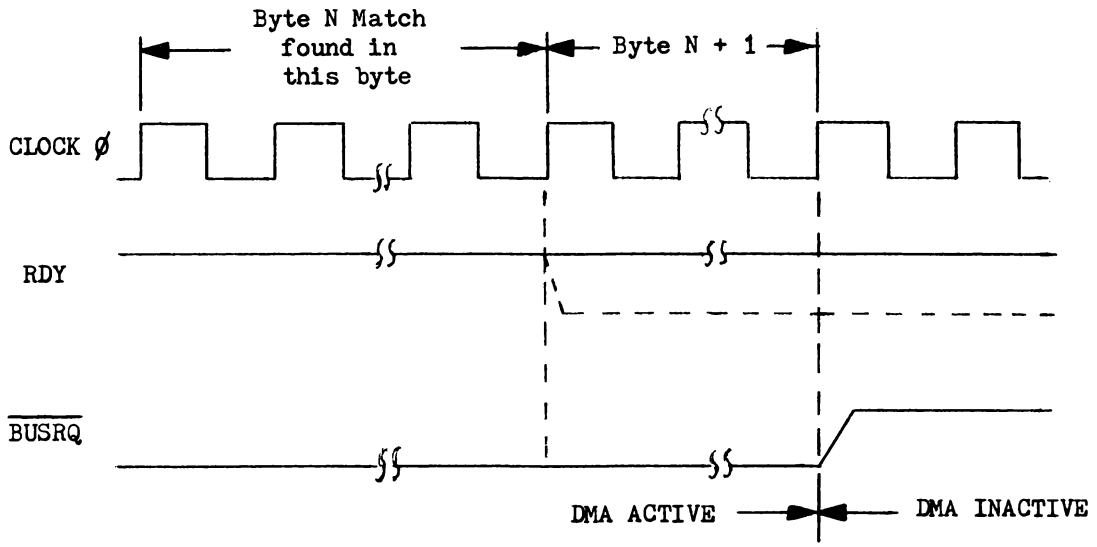


Fig. 10-14 Bus Release timing on "Match" for Burst and Continuous Modes.

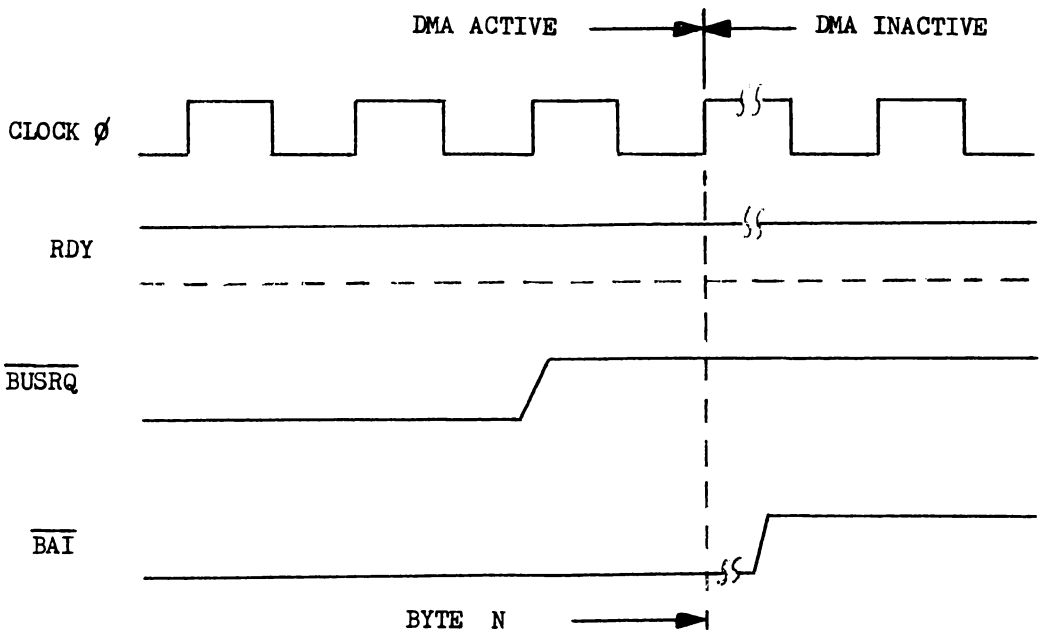


Fig. 10-15 Bus Release timing in Byte-at-a-time Mode.

10-7 REVIEW QUESTIONS

10-1 The statements below apply to the Z-80/DMA chip. Indicate true or false.

a. ___ Blocks of data can be transferred between two ports independent of the CPU.

b. ___ Transfer of data blocks can take place only between an I/O device and the main memory.

c. ___ A block of data can be searched to match a specific match byte.

d. ___ During a search operation, it is possible to transfer a block of data simultaneously.

e. ___ It is not possible to use the DMA chip in the priority daisy chain of the Z-80 system.

10-2 The following statements relate to the starting address of the DMA port(s). Circle the correct ones.

a. The starting addresses can be automatically incremented but not automatically decremented.
b. The starting address can remain fixed, if so desired.
c. The starting addresses can be automatically decremented but not automatically incremented.
d. The starting addresses can be automatically incremented as well as automatically decremented.

10-3 Fill in the blanks.

a. The address registers are _____ buffered. (Not; double; triple; quadruple)

b. The Block Length Register is _____ buffered. (Not; double; triple)

10-4 Interrupts can be generated on: (Circle the correct answers)

a. Ready
b. Match-not-found
c. End-of-block
d. Match found
e. Address found

10-5 Answer yes or not to the following questions.

a. ___ Can the DMA signal the CPU when a previously-specified number of bytes have been transferred?

b. ___ Does the transfer operation stop when the DMA signals the CPU that a previously-specified number of bytes have been transferred?

c. ___ Is it possible for the CPU to read out the status of any DMA channel upon request?

10-6 From the statements below select the one(s) that will make the following statement true.

"Under program control the DMA channel can be _____"

a. Enabled but not disabled or reset.
b. Enabled, disabled and reset.
c. Reset and enabled but not disabled.
d. Enabled and disabled but not reset.

10-7 Indicate true or false for the rates of DMA operations.

a. ___ Can be operated up to rates of 2.2 megabytes per second for the transfer operations only.

b. ___ Can be operated up to 1.25 meg abytes per second for only the search operation.

c. ___ Can be operated up to 2.2 megabytes per second for both search and transfer operations.

d. ___ Can be operated up to 1.25 megabytes for both search and transfer operations.

10-8 Under which of the following conditions can the entire DMA operation be made to repeat itself? Circle all that apply.

a. On Command (Load)
b. When the Port Address is fixed
c. Automatically (Auto Restart)
d. On End-of-Block

10-9 The following statements apply to the Byte-at-a-Time mode of operation. Indicate true or false.

a. ___ Only one byte in a data block can be transferred at any time when requested, under program control.

b. ___ The DMA retains control of the system busses after the byte transfer is completed.

c. ___ A new DMA request is made after each byte is transferred.

d. ___ Only a transfer operation can be performed in the Byte-at-a-time mode.

10-10 When operating in the Continuous mode which of the following apply if the port is not ready for the operation before End-of-Block is reached? (Circle all that apply)

a. The DMA operation continues until the End-of-Block is reached.
b. The DMA operation is terminated.
c. The DMA operation pauses until the port is ready.
d. When the port is ready it indicates this by reactivating the Ready line.

10-11 Refer to the generalized block diagram of the DMA, Fig. 10-1, and indicate true or false for the port addressing scheme.

a. ___ For data transfer operations, an 8-bit starting address is involved for ports A and B.

b. ___ The upper byte is used for addressing an I/O port.

c. ___ The single-byte address for the I/O ports is usually a fixed address.

d. ___ A 2-byte starting address is involved for both ports in memory-to-memory transfers.

10-12 Refer to Fig. 10-1, the generalized block diagram of the DMA. From the following items pick out those whose condition is indicated by the 8-bit Status Register.

a. Interrupt Pending
b. Write Address Bits Valid
c. End-of-Block
d. Byte Match found
e. Ready Active

10-13 The Bus Control and the Bus Priority Logic in Fig. 10-1 involves three request signals for control of certain Z-80 system facilities. Name these facilities.

a. _____

b. _____

c. _____

10-14 The Interrupt Control and Priority Logic block in Fig. 10-1 involves three control lines. Name the signals involved in these three lines.

a. _____

b. _____

c. _____.

10-15 The following statements relate to the 8-bit Interrupt Vector Register of Fig. 10-1. Indicate true or false.

a. ___ It holds the vector assigned to that particular DMA chip.

b. ___ The vector is preloaded during initialization of the chip.

c. ___ The contents of this register are outputted on to the data bus during the interrupt request cycle.

d. ___ The vector is sent to the CPU only if the interrupting DMA has the highest priority in the daisy chain.

e. ___ The CPU utilizes the vector for calling up the pertinent service subroutine.

10-16 Refer to Fig. 10-1 and indicate the logic block(s) in which the specific control bytes for performing the various DMA operations are loaded during chip initialization.

a. Status Register
b. Match Word Register
c. Control Logic
d. Byte Counter
e. Port Start Address Register

10-17 Refer to the Byte Counter in Fig. 10-1 and indicate true or false.

a. ___ It is an 8-bit counter.

b. ___ It is cleared at the start of the DMA operation.

c. ___ It can be either incremented or decremented.

d. ___ When the DMA processing of each byte is completed, the byte counter is incremented.

e. ___ The count in the byte counter is compared to the contents of the Match Word Register.

f. ___ The End-of-Block flag is set in the Status Register only when a match is found between the contents of the Block Length Register and the byte counter.

10-18 Fill in the blanks in the following statements which refer to the Match Word Register in Fig. 10-1.

a. It contains a word for which a match is to be found during a _____ operation. (Transfer; Search)

b. Contents of the Match Word Register are compared with the byte from the _____ _____. (Data Block; Address Register; Vector Interrupt Register)

c. When a match is found, the _____ flag in the Status Register is set. (Interrupt-Pending; End-of-Block; Ready Active; Byte-Match)

10-19 Refer to the Pulse Interval and Pulse Generation Logic block in Fig. 10-1 and indicate true or false.

a. ___ The 16-bit Pulse Control Register is loaded during DMA chip initialization.

b. ___ The length of the data block, expressed in number of bytes, is loaded into the Pulse Control Register.

c. ___ The DMA emits a signal only when the entire block is processed.

d. ___ The signal emitted by the DMA is interpreted as an interrupt signal by the CPU.

e. ___ The signal emitted by the DMA is used to communicate with a peripheral device.

10-20 The statements below apply to the Mask Word Register in Fig. 10-1. Fill in the blanks.

a. This register is loaded during _____ . (Initialization; Start of the data block; End of the data block)

b. The word in the register masks out certain bits loaded in the _____ _____ Register. (Block length; Byte Counter; Start Address; Interrupt Vector; Match Word; Pulse Control)

10-21 Which of the following signals requests the CPU to release the system busses?

a. $\overline{\text{BAI}}$
b. $\overline{\text{BUSRQ}}$
c. $\overline{\text{BAO}}$
d. $\overline{\text{M1}}$

10-22 After the DMA has obtained control of the system busses, it outputs a signal on the _____ pin. ($\overline{\text{MREQ}}$: $\overline{\text{RD}}$; $\overline{\text{IORQ}}$; $\overline{\text{WR}}$; $\overline{\text{M1}}$)

10-23 Which of the following signals(s) can be used to slow down the DMA if the I/O devices and/or memory are slower than the DMA chip?

a. $\overline{\text{RDY}}$
b. $\overline{\text{CE/WAIT}}$
c. $\overline{\text{MREQ}}$
d. $\overline{\text{IORQ}}$
e. $\overline{\text{BUSRQ}}$

10-24 The following statements apply to the Search operations. Indicate true or false.

a. ___ With 2.5 MHz Z-80/DMA, search rates of up to 2.0 M bytes per second are possible.

b. ___ With 4.0 MHz Z-80/DMA, search rates of up to 2.0 M bytes per second are possible.

c. ___ With 2.5 MHz Z-80/DMA, search rates of up to 4.0 M bytes per second are possible.

d. ___ With 4.0 MHz Z-80/DMA, search rates of up to 4.0 M bytes per second are possible.

10-25 Which of the following transfers are possible when the DMA is programmed in the Transfer class of operations?

a. CPU-to-memory transfers.
b. CPU-to-I/O transfers.
c. Memory-to-I/O transfers.
d. I/O-to-CPU transfers.
e. Memory-to-Memory transfers.

10-26 The following statements apply to the Auto Restart feature of the Z-80/DMA. Indicate true or false.

a. ___ The Byte Counter is not affected when the addresses are loaded.

b. ___ The starting addresses for either port is reloaded when an interrupt comes along.

c. ___ A control bit in one of the command words activates Auto Restart.

d. ___ An Auto Restart can start a block transfer or search operation at a new starting address.

10-27 Refer to Fig. 10-4, Variable Cycle Length timing. Which of tne following control signals can be terminated one-half T-cycle early?

a. $\overline{\text{BUSRQ}}$
b. $\overline{\text{MREQ}}$
c. $\overline{\text{INT}}$
d. $\overline{\text{CE/WAIT}}$
e. $\overline{\text{IORQ}}$
f. $\overline{\text{BAI}}$
g. $\overline{\text{BAO}}$

10-28 Which of the following signals must be simultaneously active for read-out of the DMA Status Register?

a. $\overline{WR}$
b. $\overline{BAO}$
c. $\overline{IORQ}$
d. $\overline{RD}$
e. $\overline{CE}$
f. $\overline{BAI}$

10-29 Refer to the Memory-to-I/O Transfer timing of Fig. 10-7. Answer the following questions. (Yes or no)

a. ___ Is it possible to include a TW state in the Memory Read cycle?

b. ___ Is it possible to include a second TW state in the I/O write cycle?

c. ___ Is it possible to insert a TW state between T1 and T2 of the I/O write cycle?

d. ___ Is the TW state in the I/O write cycle inserted automatically?

10-30 For proper control of the I/O-to-I/O transfers (Fig. 10-10). Certain control signals must be simultaneously active.

a. _____ Which pair must be simultaneously active for the I/O write operations? ($\overline{RD}$; $\overline{WR}$; $\overline{BAI}$; $\overline{MREQ}$; $\overline{IORQ}$; $\overline{CE/WAIT}$)

b. _____ Which signals must be simultaneously active for the I/O read operation? ($\overline{RD}$; $\overline{WR}$; $\overline{BAI}$; $\overline{MREQ}$; $\overline{IORQ}$; $\overline{CE/WAIT}$)

10-31 For all 3 modes of operation, for the DMA to be active it must have control of the system busses. Which signals must be simultaneously active for Bus Acceptance and Release?

a. $\overline{BAI}$, $\overline{RD}$ and $\overline{IORQ}$

b. RDY, $\overline{IORQ}$ and $\overline{BUSRQ}$

c. $\overline{BAI}$, RDY and $\overline{BUSRQ}$

d. $\overline{BAO}$, $\overline{BUSRQ}$, $\overline{IORQ}$ and RDY

11 PROGRAMMING THE DMA CHIP

CHAPTER OBJECTIVES

The objectives of this chapter are as follows:

1. To introduce the concept of customizing the DMA chip for specific applications involving transfer, search and combined transfer-search operations.

2. To present the command bytes for the basic set-up of DMA Port A, together with the four associated registers.

3. To present the variable timing byte for Port A.

4. To describe the basic set-up for Port B together with the variable timing byte.

5. To examine the format of the command byte for the search class of operations, including the match and mask bytes.

6. To present the command byte for the pulse generation, interrupt generation and the interrupt vector features of the DMA along with its five associated registers.

7. To present the command byte for miscellaneous signals associated with control of the DMA operation.

8. To describe the command byte for several other functions such as resetting, enabling and disabling several features of the DMA chip.

9. To present the features of the Read Mask Register and the Status Register.

TEXTBOOK REFERENCES

For reviewing material in the main hardcover textbook, relevant to the topics in this chapter, the following chapter(s) and/or sections are suggested:

1. For general discussion of DMA - Chapter 10

2. For transparent mode - Sec. 10-2.1

3. For cycle stealing mode - Sec. 10-2.2

4. For Byte-at-a-time DMA Transfers,

   a) Using single memory - Sec. 10-3.1
   b) Using multiple memories - Sec. 10-3.2

5. For Continuous Mode single-block transfers - Sec. 10-4

6. For transfer of multiple data blocks - Sec. 10-5

7. For CPU response to DMA request - Sec. 10-6.2

8. For operation of the DMA interface chip - Sec. 10-7

9. For byte and block count - Sec. 10-7.3

10. For byte count update - Sec. 10-7.7

11. For block count update - Sec. 10-7.8

11-1 INTRODUCTION

The Z-80/DMA is a programmable interface chip which, under program control, can be initialized to perform functions for specific applications within certain limits. Customizing the DMA requires that the chip be first set up by writing programming or command bytes into some or all of the seven groups of write registers. The DMA chip has two ports and either a memory or an I/O device can be connected to either of the two ports. Transfers in either direction are possible between the two ports. It is also possible to search a data block for a match with a pre-established match byte whose bits can be selectively masked out, if so desired. In this chapter we describe the initialization process and the command bytes involved in a set-up program. Also, examples are included that show how the DMA is initialized to perform transfer, search and combined transfer-search operations.

11-2 COMMAND WORDS FOR DMA SET-UP

In this section we discuss the formats of the various command words which are required to initialize the DMA chip. The words or bytes define the different classes and modes of the DMA operation. The DMA chip has seven groups of Write Registers. Each group consists of a base register. Some of these groups have one or more associated write registers. The seven groups are labelled WR0 - WR6 and there is a total of 14 associated registers. The procedure of writing into any of the register groups which has one or more associated register(s) consists of the following steps.

1. The desired word is first written into the base register,

2. Appropriate pointer bits are included in the byte written into the base register, and

3. The pointer then sets up the address for each of the sequentially associated registers and the bytes are then successively written into them.

Each of the seven write register groups, and their associated registers, are briefly described below. Remember that the DMA chip has two fundamental states which can be programmed. In the enabled state, the DMA obtains control of the system busses and thereby controls the transfer of data between the addressed ports. In the disabled state, no requests for busses are made and so no data are transferred between ports. When power is first applied to the DMA chip it automatically goes into the disabled state. It also goes into the disabled state when the chip is reset by any means. It is possible to write command bytes into the write registers, in either of the two states but the writing process immediately puts the chip into the disabled state. It remains in that state until an enable command is sent by the CPU.

11-2.1 The WR0 Group

This group which contains the basic set-up information for the DMA chip consists of one base register WR0-0 and four associated registers labelled WR0-1 through WR0-4. Figure 11-1 shows these registers and what is contained in each. The arrows show the bit in the base register which identifies or affects the associated registers.

Bits D0 and D1 in the base register identify the class of operation as in the following codes:

C_1	C_2	OPERATION CLASS
0	0	Not used
0	1	Data block transfer only
1	0	Search only
1	1	Combined Search/transfer

Bit position D2 in the base register identifies which port is the source port. Selecting one port automatically makes the other port the destination port for transfer operations. If the Search only mode is selected, then D2 identifies only the source port. No destination port is addressed or selected.

11-2.2 The WR1 Group

This group also contains the basic set-up information for Port A of the DMA chip. It consists of the base register WR1-0 and one associated register WR1-1 as shown in Fig. 11-2.

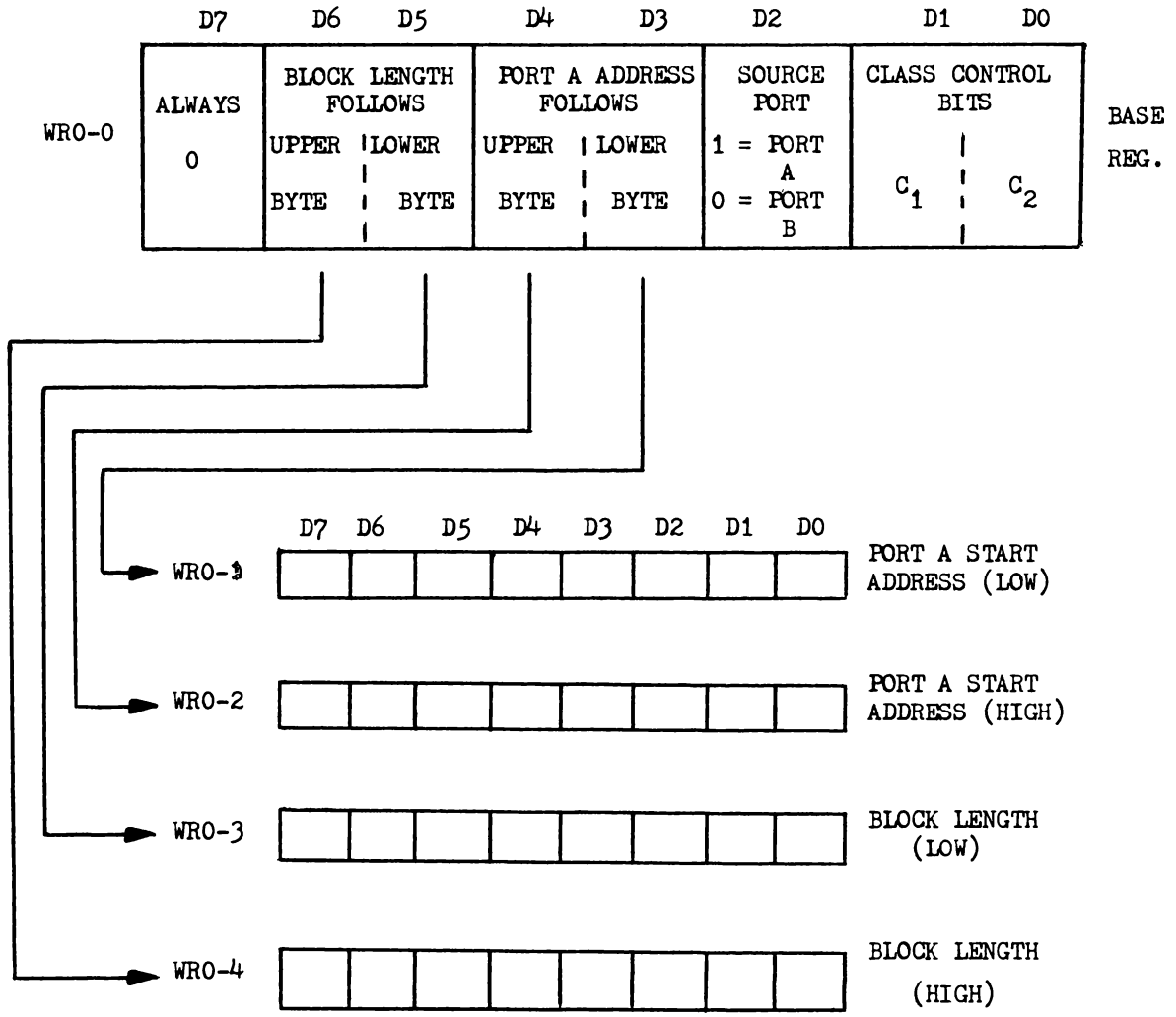


Fig. 11-1 Write registers group WR0.

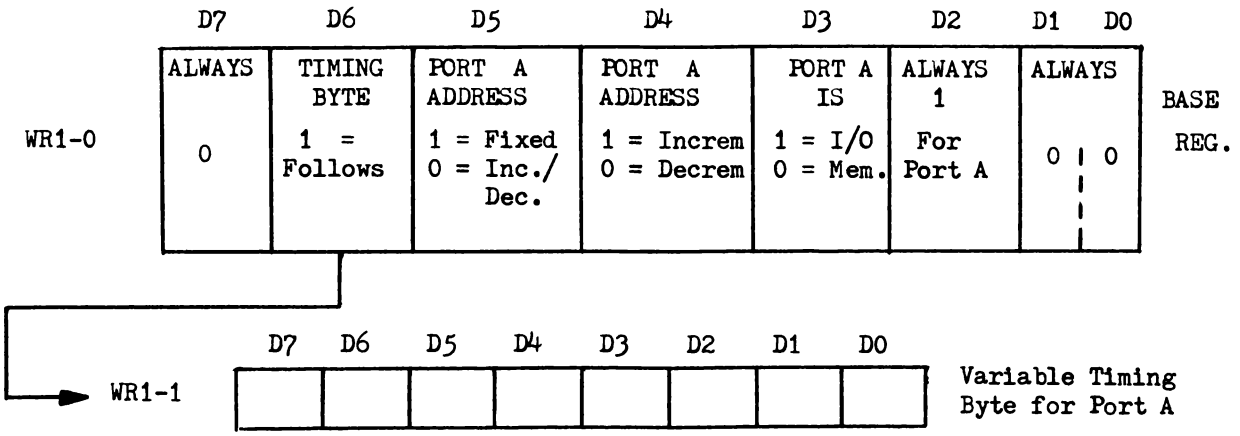


Fig. 11-2 Write registers group WR1.

For the Base Register WR1-0

1. A 1 in bit D5 indicates a fixed address for Port A and so the bit in D4 has no effect on this operation. It can either be a 0 or a 1.

2. A 0 in bit D5 indicates that the Port A address can either be incremented or decremented after each byte transfer, as indicated by the bit in D4 position.

For the Variable Timing Register

1. Bits D4 and D5 are always 0.

2. The variable cycle length time is indicated by bits D0 and D1 as follows:

D1	D0	No. of T-Cycles
0	0	4
0	1	3
1	0	2
1	1	Not used

3. A 0 in the remaining bit positions indicates the following timing changes. (Note that these timing variations are previously explained in Chapter 10)

0 in Bit Position	Ends ½ cycle early
D2	$\overline{\text{IORQ}}$
D3	$\overline{\text{MREQ}}$
D6	$\overline{\text{RD}}$
D7	$\overline{\text{WR}}$

11-2.3 The WR2 Group

This group contains the basic set-up information for Port B of the DMA chip. It consists of the base register WR2-0 and one associated register WR2-1 as shown in Fig. 11-3. The register formats are identical to that for Port A as shown in Section 11-2.2.

For the Base Register WR2-0

1. A 1 in the bit D5 position indicates a fixed address for Port B and so the bit in D4 has no effect on the operation. It can be either a 0 or a 1.

2. A 0 in bit D5 position indicates that the Port B address can either be incremented or decremented after each byte transfer, as indicated by the bit in D4 position.

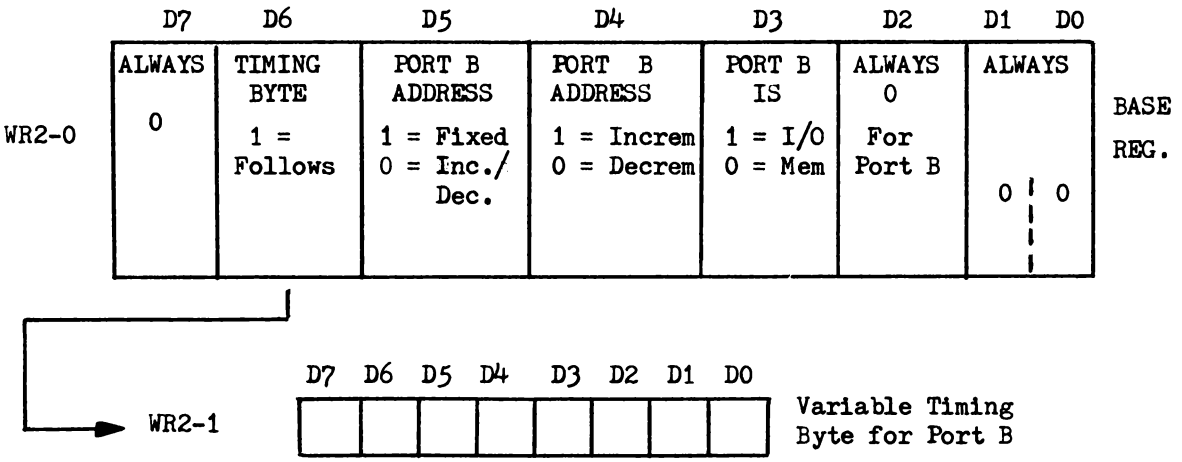


Fig. 11-3 Write registers group WR2.

For the Variable Timing Register WR2-1

1. Bits D4 and D5 are always 0.

2. The variable cycle length time is indicated by bits D0 and D1 as follows:

D1	D0	No. of T-Cycles
0	0	4
0	1	3
1	0	2
1	1	1

3. A 0 in the remaining bit positions indicate the following timing changes. (Note that these timing variations are previously explained in Chapter 10.)

0 in Bit Positions	Ends ½ cycle early
D2	$\overline{\text{IORQ}}$
D3	$\overline{\text{MREQ}}$
D6	$\overline{\text{RD}}$
D7	$\overline{\text{WR}}$

11-2.4 The WR3 Group

This group contains information for the Search Class of operations. It consists of the base register WR3-0 and two associated registers WR3-1 and WR3-2 for the Mask and the Match bytes respectively. The register formats are shown in Fig. 11-4.

In the Mask byte, WR3-1, a 0 in any bit position permits the bit in the corresponding bit position in the Match byte (WR3-2) to be compared. A 1 in any bit position of WR3-1 masks out the corresponding bit in WR3-2.

11-2.5 The WR4 Group

This group contains one base register, WR4-0, and five associated registers, WR4-1 through WR4-5. This group of commands is concerned with the Pulse generation, Interrupt generation and the Interrupt Vector features of the Z-80/DMA. The register formats are shown in Fig. 11-5.

For the Base Register WR4-0

1. The base register provides pointers in positions D2 and D3 which indicate that the following two bytes are the block starting address for Port B. D2 points to the low byte (WR4-1) and D3 to the high point (WR4-2). This is done by means of a 1 in both D2 and D3.

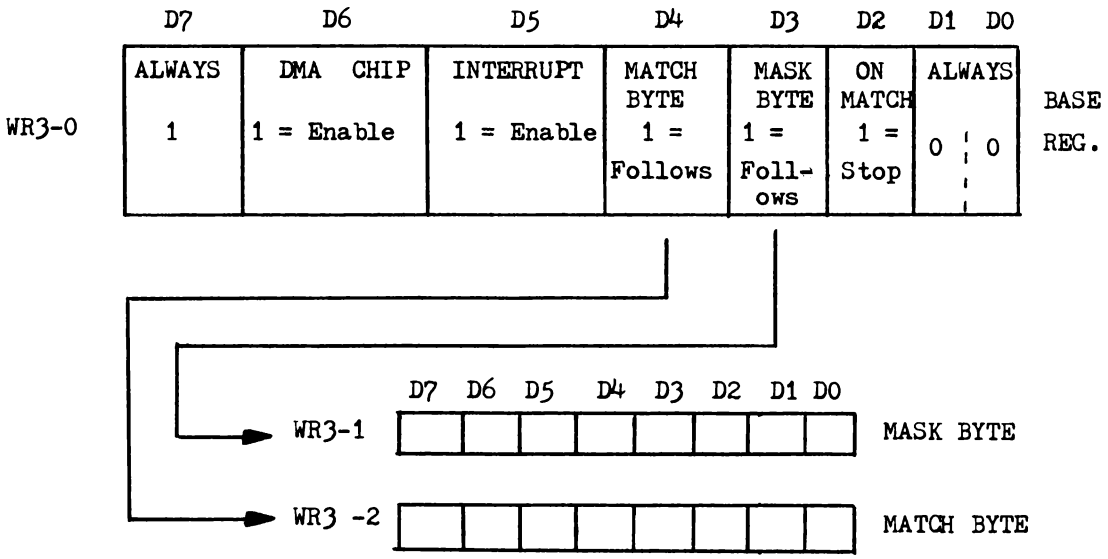


Fig. 11-4 Write registers group WR3.

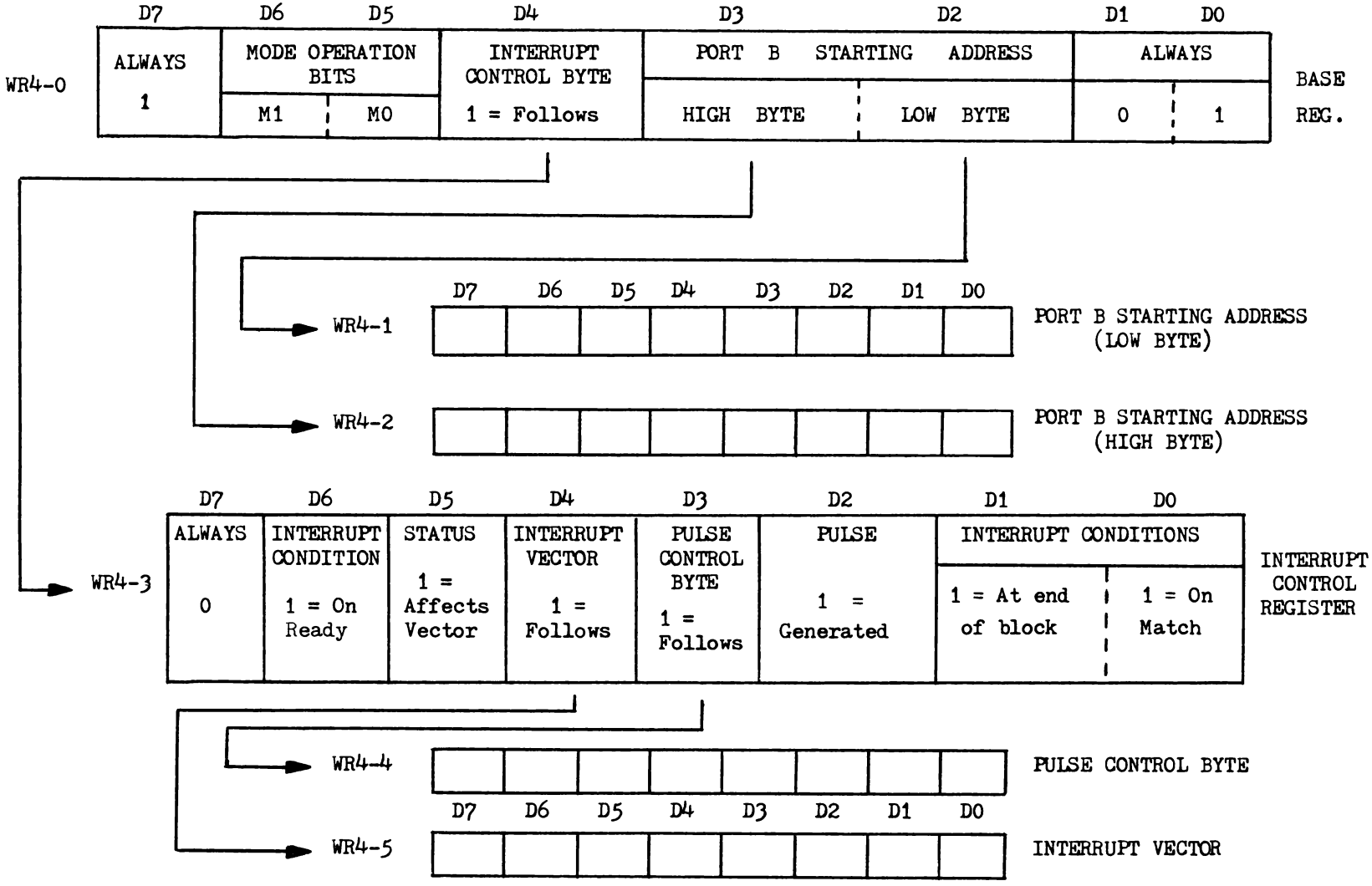


Fig. 11-5 Write registers group WR4.

2. The Mode of Operation is defined by bits in positions D5 and D6 as follows:

M1	M0	Mode of Operation
0	0	Byte-at-a-time
0	1	Continuous
1	0	Burst
1	1	Transparent

For the Interrupt Control Byte WR4-3

1. Bit D7 is always a 0.

2. A 1 in the following bit positions defines the conditions under which an interrupt is generated:

Bit Positions	Interrupt Generating Condition
D0	On Match
D1	At the end of the Data Block
D6	On Ready

3. A 1 in D2 generates a pulse on the $\overline{\text{INT}}$ terminal line when the entire data block is processed. The length of the block, in bytes, is loaded in the Pulse Control Byte Register WR4-4.

4. A 1 in D3 indicates that a Pulse Control Byte follows.

5. A 1 in D4 indicates that the Interrupt Vector follows and is to be loaded in register WR4-5.

6. A 1 in D5 indicates that the Status Register's contents will determine and modify the Interrupt Vector Byte (WR4-5) as shown below. (Note that the format of the Status Byte is described later in Sec. 11-2.8.)

For the Pulse Control Byte WR4-4

1. An 8-bit control word is loaded into WR4-4. This is usually the desired number of bytes in the data block.

2. Upon completion of each DMA byte operation, either a transfer or a Search, the contents of this register are compared with the contents of the low-order byte of the byte counter.

3. When the two bytes compare, i.e. they are the same, a pulse is generated on the $\overline{\text{INT}}$ line. However, a legitimate interrupt is not generated.

For the Interrupt Vector Byte WR4-5

1. The contents of this register identify the particular DMA chip in the Z-80 system.

2. The byte in WR4-5 is transferred to the CPU during the Interrupt Acknowledge provided the interrupting DMA is the highest priority device on the daisy chain.

3. When bit D5 of the Interrupt Control Byte (WR4-3) is a 1, and the DMA chip is programmed to generate an interrupt when a specific status condition exists, the vector in WR4-5 will be changed as shown below by modifying bits D1 and D2.

D2	D1	Status Condition
0	0	Interrupt on Ready - RDY
0	1	Interrupt on Match
1	0	Interrupt on End-of-Block
1	1	Interrupt on Match and End-of-Block

11-2.6 The WR5 Group

This group contains only the base register WR5-0 and has no associated registers. It contains miscellaneous signals associated with control of the DMA operation. The format of this byte is shown in Fig. 11-6 and the functions performed by each bit are presented below.

	D7	D6	D5	D4	D3	D2	D1 D0	
WR5-0	ALWAYS 1	NOT USED 1 or 0	END OF BLOCK 0 = STOP 1 = AUTO RESTART	PIN #16 0 = $\overline{CE}$ ONLY 1 = $\overline{CE}$ and $\overline{WAIT}$ Multiplexed on same pin.	READY 1 = ACTIVE HIGH 0 = ACTIVE LOW	NOT USED 1 or 0	ALWAYS 1 | 0	BASE REG.

Fig. 11-6 Write register group WR5.

1. Bit positions D2 and D6 are not used and so they can be either a 0 or a 1.

2. Bit position D3 indicates whether the input on the RDY pin (pin #25) is to be interpreted as active when it is high or low.

3. Bit position D4 specifies whether the CE/WAIT input pin (pin #16) is to be interpreted as a chip enable or as a wait during the time when the bus acknowledge signal input is active.

4. Bit position D5 indicates the action to be taken when the end of block is reached. A 0 in D5 stops the DMA operation and a 1 is automatically repeats the entire operation.

11-2.7 The WR6 Group

This group contains the base register WR6-0 and one associated register WR6-1. This group performs a number of miscellaneous functions including resetting, as well as enabling and disabling various features. The formats of the two registers are shown in Fig. 11-7.

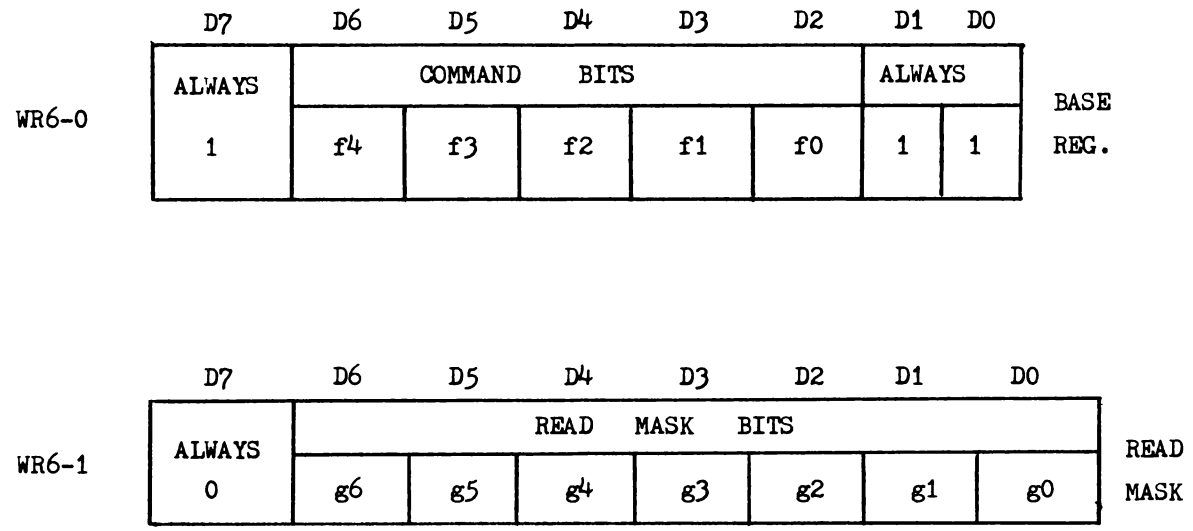


Fig. 11-7 Write register group WR6.

For the Base Register WR6-0

1. With the exception of bit positions D0, D1 and D7 which always contain a 1, positions D2-D6 are coded F0 - F4. They contain the applicable binary codes.

2. The F0 - F4 codes and the functions they perform are given below in Table 11-1. For programming convenience the corresponding HEX codes are also included.

Table 11-1 Command codes for register WR6-0.

HEX CODE	COMMAND BITS					COMMAND DESCRIPTION
	f4	f3	f2	f1	f0	
C3	1	0	0	0	0	Resets Interrupt circuitry, disables Interrupt and Bus Request logic, removes Internal Ready condition, disables "MUXCE" and stops Auto Repeat.
C7	1	0	0	0	1	Resets Port A timing to standard Z-80 timing.
CB	1	0	0	1	0	Resets Port B timing to standard Z-80timing.
CF	1	0	0	1	1	Loads starting address for Ports A and B, and clears the Byte Counter.
D3	1	0	1	0	0	Continues address from present location and clears the Byte Counter.
AB	0	1	0	1	0	Enables Interrupts.
AF	0	1	0	1	1	Disables Interrupts.
A3	0	1	0	0	0	Resets and disables Interrupt circuitry like (RETI) and removes or unforces the internal Ready condition.
87	0	0	0	0	1	Enables DMA (See * note below).
83	0	0	0	0	0	Disables DMA (See ** note below).
A7	0	1	0	0	1	Initiates Read Sequence to the first register designated as readable by the Read Mask Register WR6-1.
BF	0	1	1	1	1	Sets Read Status so that the next readout is from the Status Register.
B3	0	1	1	0	0	Forces an internal Ready condition regardless of the input on the RDY pin. (See ** note below).
8B	0	0	0	1	0	Clears Match and End-of-Block Status bits.
B7	0	1	1	0	1	Enable after RETI instruction so that the DMA requests bus only after it receives a RETI. An enable DMA command must follow this code.
BB	0	1	1	1	0	A Read Mask byte follows.

Notes

* Both codes 87 and 83 Enable and Disable DMA respectively, affect all operations except interrupts. They do not reset any functions.

** This command (B3) is used for Memory-to-Memory operations where a RDY signal is not required. This command does not function in a Byte-at-a-time Mode.

For the Read Mask Register WR6-1

1. If the BB code is used in the base register WR6-0, then WR6-1 must follow.

2. The bit in the D7 position is always a 0.

3. A 1 in bit position D0 - D6 enables the read out of the various register bytes as shown below.

   D0 - Status Register (RR0)
   D1 - Byte Counter, low byte (RR1)
   D2 - Byte Counter, high byte (RR2)
   D3 - Port A Address, low byte (RR3)
   D4 - Port A Address, high byte (RR4)
   D5 - Port B Address, low byte (RR5)
   D6 - Port B Address, high byte (RR6)

11-2.8 The Status Register

The seven registers, identified above as RR0 - RR6, are the read registers. When the bit position D0 of the Mask Register WR6-1 contains a 1, the contents of the Status Register are read out. The format of the Status Register is shown below in Fig. 11-8. Note that the status bits are active low.

D7	D6	D5	D4	D3	D2	D1	D0
NOT USED		END OF BLOCK	MATCH FOUND	INTERRUPT PENDING	NOT USED	READY ACTIVE	DMA OPERATION HAS OCCURED
0 or 1	0 or 1	0	0	0	0 or 1	0	1

Fig. 11-8 Status register.

11-3 PROGRAMMING THE DMA FOR TRANSFER ONLY OPERATIONS

In this section we show, by means of typical examples, how the DMA chip can be programmed for the transfer-only class of operations, which allows us to transfer blocks of data between the source and the destination ports. (See Sec. 10-5.1 for review of this class of operation.)

11-3.1 I/O-to-Memory Transfer

Example 11-1

A peripheral device whose starting address is 36_H is connected to Port A and a memory is connected to Port B. A block of data whose length is 0435_H bytes is to be transferred from the peripheral to the memory at starting address 1820_H using the Continuous Mode of transfer. An interrupt must be generated on the Ready condition and the $\overline{INT}$ line is to be pulsed every time byte 87_H of the block is transferred. The Interrupt Vector of the peripheral is $4E_H$. The Ready signal is active high and the $\overline{CE}$ pin is interpreted only as a Chip Enable.

Design the DMA initialization program for this problem giving each byte in both machine language code and the hexadecimal code and identify each write register (WR) used, along with a brief explanation of each byte.

Solution

The solution for this problem is given in convenient tabular form in Table 11-2. In step 13 of this table note that an X and a 0 are indicated in bit positions D2 and D6. The X means that these bit positions do not affect the operation of that particular register. However, for purposes of giving the proper hexadecimal code, either a 1 or a 0 must be used in these two positions. Arbitrarily we have chose to use a 0 in both these bit positions.

Table 11-2 DMA initialization byte for example 11-1

BYTE NO.	WRITE REGISTERS		FUNCTIONS AND COMMENTS	MACHINE LANGUAGE CODE								HEX CODE
	BASE	ASSOCIATED		D7	D6	D5	D4	D3	D2	D1	D0	
1	WR0-0	-	Setup byte - direction of transfer, Port A address and block block length follows.	0	1	1	0	1	1	0	1	6D
2	-	WR0-1	Port A start address - low byte.	0	0	1	1	0	1	1	0	36
3	-	WR0-3	Block length - low byte	0	0	1	1	0	1	0	1	35
4	-	WR0-4	Block length - high byte	0	0	0	0	0	1	0	0	04
5	WR1-0	-	Port A is peripheral with fixed address. No variable timing byte follows	0	0	1	1	1	1	0	0	3C
6	WR2-0	-	Port B setup byte. Port B is memory. Starting address is incremented. No variable timing byte follows.	0	0	0	1	0	0	0	0	10
7	WR4-0	-	Defines Continuous Mode of operation. Port B starting address bytes and interrupt control byte follows.	1	0	1	1	1	1	0	1	BD
8	-	WR4-1	Port B start address - low byte	0	0	1	0	0	0	0	0	20
9	-	WR4-2	Port B start address - high byte	0	0	0	1	1	0	0	0	18
10	-	WR4-3	Interrupt Control byte - INT is generated on Ready condition.	0	1	0	1	1	1	0	0	5C
11	-	WR4-4	Pulse Control Byte - compared with Byte Counter - generates pulse on $\overline{INT}$ line when the two compare.	1	0	0	0	0	1	1	1	87
12	-	WR4-5	Interrupt Vector of peripheral.	0	1	0	0	1	1	1	0	4E
13	WR5-0	-	Control byte - ready active high, stop End-of-Block, and $\overline{CE}$ is chip enable only.	1	0 X	0	0	1	0 X	1	0	8A
14	WR6-0	-	Load starting addresses and reset Byte Counter.	1	1	0	0	1	1	1	1	CF
15	WR6-0	-	Enable DMA.	1	0	0	0	0	1	1	1	87

11-3.2 Memory-to-Memory Transfers

Example 11-2

One block if data, consisting of 128_{10} bytes, is to be transferred from memory Y (connected to Port A) to memory Z (connected to Port B). The starting address of the block in memory Y is $3F94_H$ and each subsequent address is decremented. The starting address of the block in memory Z is $28AA_H$ and each subsequent address is decremented. The Burst Mode is used. Upon completion of the block transfer, a pulse is to be generated on the $\overline{INT}$ line. The $\overline{CE}$ is used only as a chip enable. Memory Z is slower than memory Y and requires two T-cycles. The Interrupt Vector of memory Y (Port A) is $7B_H$. Ready is indicated as active low and the DMA activity is terminated at the end of the block.

Design the DMA initialization program for this problem, identifying each write register (WR) along with a brief explanation of each byte. Give both the machine language and the hexadecimal codes for each byte.

Solution

The solution for this problem is given in convenient tabular form in Table 11-3. In step 13 of this table note that an X and a 0 are indicated in bit positions D2 and D6. The X means that these bit positions do not affect the operation of that particular register. However, for purposes of giving the proper hexadecimal code, either a 1 or a 0 must be used in these two positions. Arbitrarily we have chosen to use a 0 in both these two positions.

Table 11-3 DMA initialization bytes for example 11-2

BYTE NO.	WRITE REGISTERS BASE	WRITE REGISTERS ASSOCIATED	FUNCTIONS AND COMMENTS	MACHINE LANGUAGE CODE D7	D6	D5	D4	D3	D2	D1	D0	HEX CODE
1	WR0-0	-	Setup byte - Port A address, direction of transfer and block length follows.	0	0	1	1	1	1	0	1	3D
2	-	Wr0-1	Port A start address - low byte.	1	0	0	1	0	1	0	0	94
3	-	WR0-2	Port A start address - high byte.	0	0	1	1	1	1	1	1	3F
4	-	WR0-3	Block length - low byte.	1	0	0	0	0	0	0	0	80
5	WR1-0	-	Port A is memory - addresses are not fixed - Port A address increments. Variable timing byte does not follow.	0	0	0	1	0	1	0	0	14
6	WR2-0	-	Port B setup byte - Port B is memory, starting address is decremented. Vairable timing byte follows.	0	1	0	0	0	0	0	0	40
7	-	WR2-1	Variable cycle length time - 2 additional T cycles - $\overline{IORQ}$, $\overline{MREQ}$, $\overline{RD}$ and $\overline{RW}$ do not end $\frac{1}{2}$ cycle early.	1	1	0	0	1	1	1	0	CE
8	WR4-0	-	Defines Burst Mode of operation. Port B starting address bytes and interrupt control byte follows.	1	1	0	1	1	1	0	1	DD
9	-	WR4-1	Port B start address - low byte.	1	0	1	0	1	0	1	0	AA
10	-	WR4-2	Port B start address - high byte	0	0	1	0	1	0	0	0	28
11	-	WR4-3	Interrupt Control Byte - Interrupt is generated on Ready and End-of-Block - Pulse Control byte does not follow.	0	1	1	1	0	0	1	0	72
12	-	WR4-5	Interrupt Vector of Memory Y (Port A).	0	1	1	1	1	0	1	1	7B
13	WR5-0	-	Control byte - ready active low, $\overline{CE}$ is chip enable only, and stop DMA at End-of Block.	1	0 X	0	0	0	0 X	1	0	82
14	WR6-0	-	Load starting addresses and reset Byte Counter.	1	1	0	0	1	1	1	1	CF
15	WR6-0	-	Enable DMA.	1	0	0	0	0	1	1	1	87

11-4 PROGRAMMING THE DMA FOR SEARCH OPERATIONS

Example 11-3

A block of data, 96_{10} bytes long, is stored in memory starting at address $2A2A_H$. The block addresses increment from this address. The memory is connected to Port A. The data block is to be searched for a match on $8D_H$. The match must be obtained on all bits of this byte. The DMA operation must stop when a match is obtained and an interrupt must be generated. The operation must then branch to a subroutine whose starting address low byte is BC_H. The Byte-at-a-Time Mode of operation is used. Ready is active high and the $\overline{CE}$ pin is used for chip enable only.

Design the DMA initialization program for this problem, identifying each write register (WR) along with a brief explanation of each byte. Give both the machine language and the hexadecimal codes for each byte.

Solution

The solution for this problem is given in convenient tabular form in Table 11-4. In step 10 of this table note that an X and a 0 are indicated in bit positions D2 and D6. The X means that these bit positions do not affect the operation of that particular hexidecimal code, either a 1 or a 0 must be used in these two positions. Arbitrarily we have chosen to use a 0 in both these bit positions.

Table 11-4 DMA initialization bytes for example 11-3.

BYTE NO.	WRITE REGISTERS BASE	WRITE REGISTERS ASSOCIATED	FUNCTIONS AND COMMENTS	MACHINE LANGUAGE CODE D7	D6	D5	D4	D3	D2	D1	D0	HEX CODE
1	WR0-0	-	Setup byte - Port A address - block length follows	0	0	1	1	1	1	1	0	3E
2	-	WR0-1	Port A start address - low byte.	0	0	1	0	1	0	1	0	2A
3	-	WR0-2	Port A start address - high byte.	0	0	1	0	1	0	1	0	2A
4	-	WR0-3	Block length - low byte.	0	1	1	0	0	0	0	0	60
5	WR1-0	-	Port A is memory - address is not fixed - address decrements - no variable timing byte follows.	0	0	0	1	0	1	0	0	14
6	WR3-0	-	Search class of operation - match byte follows, mask byte does not follow - stop DMA on match and generate interrupt.	1	0	1	1	0	1	0	0	B4
7	-	WR3-2	Match byte (all bits).	1	0	0	0	1	1	0	1	8D
8	WR4-0	-	Defines Byte-at-a-Time Mode - no Port B address - interrupt control byte follows.	1	0	0	1	0	0	0	1	91
9	-	WR4-3	Interrupt Control Byte - interrupt generated on match - Interrupt Vector follows (loaded in WR4-5).	0	0	0	1	0	0	0	1	11
10	-	WR4-5	Interrupt Vector of Memory (Port A)	1	0	1	1	1	1	0	0	BC
11	WR5-0	-	Control byte - ready active low - $\overline{CE}$ is chip enable only.	1	0	0	0	1	0	1	0	8A
12	WR6-0	-	Enable DMA.	1	0	0	0	0	1	1	1	87

11-5 PROGRAMMING THE DMA FOR SEARCH-TRANSFER OPERATIONS

Example 11-4

Memory A is connected to Port A of the DMA chip and memory B to Port B. A block of data, $B2_H$ in length is to be transferred from memory B to memory A. The starting address in memory A is $24DD_H$ and $E923_H$ in memory B. Both addresses are decremented. Simultaneously, a match is to be performed on bits D0, D3, D4 and D7 only of the byte 77_H. When a match is found, an interrupt must be generated and a service subroutine, whose starting address low byte is 99_H, is to be executed. If a match is not found, then the DMA operation must be terminated upon completion of the block transfer. The $\overline{CE}$ pin is used for chip enable only. Byte-at-a-Time mode of operation is used.

Design the DMA initialization program for this problem, identifying each write register (WR) along with a brief explanation of each byte. Give both the machine language and the hexadecimal codes for each byte.

Solution

The solution for this problem is given in convenient tabular form in Table 11-5.

Table 11-5 DMA initialization bytes for example 11-4

BYTE NO.	WRITE REGISTERS		FUNCTIONS AND COMMENTS	MACHINE LANGUAGE CODE								HEX
	BASE	ASSOCIATED		D7	D6	D5	D4	D3	D2	D1	D0	CODE
1	WR0-0	-	Setup byte - Port A address, defines Port B as source - block length follows.	0	0	1	1	1	0	1	1	3B
2	-	WR0-1	Port A start address - low byte.	1	1	0	1	1	1	0	1	DD
3	-	WR0-2	Port A start address - high byte.	0	0	1	0	0	1	0	0	24
4	-	WR0-3	Block length - low byte.	1	0	1	1	0	0	1	0	B2
5	WR1-0	-	Port A is memory - address is not fixed - address decrements Variable timing byte does not follow.	0	0	0	0	0	1	0	0	04
6	WR2-0	-	Setup byte for Port B - address decrements - Port B is memory - variable timing byte does not follow.	0	0	0	0	0	0	0	0	00
7	WR3-0	-	Defines Search class of operation - match and mask bytes follow - stop on match and generate interrupt.	1	0	1	1	1	1	0	0	BC
8	-	WR3-1	Mask byte - bits D1, D2, D5 and D6 are masked	0	1	1	0	0	1	1	0	66
9	-	WR3-2	Match byte	0	1	1	1	0	1	1	1	77
10	WR4-0	-	Defines Byte-at-a-Time Mode - Port B address and interrupt control bytes follow.	1	0	0	1	1	1	0	1	9D
11	-	WR4-1	Port B start address - low byte.	0	0	1	0	0	0	1	1	23
12	-	WR4-2	Port B start address - high byte.	1	1	1	0	1	0	0	1	E9
13	-	WR4-3	Interrupt Control Byte - interrupt generated on match - Interrupt Vector follows.	0	0	0	1	0	0	0	1	11
14	-	WR4-5	Interrupt Vector	1	0	0	1	1	0	0	1	99
15	WR5-0	-	Control byte - Pin #16 used for $\overline{CE}$.	1	0	0	0	0	0	1	0	82
16	WR6-0	-	Enable DMA	1	0	0	0	0	1	1	1	87

11-6 REVIEW QUESTIONS

11-1 The following statements apply to the write registers used for initialization of the DMA chip. Fill in the blanks.

a. There are _____ base registers and a total of _____ associated registers. (2; 4; 6; 7; 8; 12; 14; 16)

b. Pointer bits are written into the _____ registers. (Base; associated)

c. Control bytes are successively written into the _____ registers. Base; associated)

11-2 The Base Register of the WR0 group is loaded with the following byte:

0 0 1 1 1 1 1 0

Answer the following:

a. ___ Which port is the source port? (A; B)

b. ___ How many bytes are involved in the source port address? (1; 2)

c. ___ Which class of operation is the DMA programmed for? (Transfer only; Search only; Combined Search/Transfer)

d. ___ How many bytes does the number specifying the Block Length contain? (1; 2)

11-3 The following statements apply to the WR1 group whose registers are loaded as shown below. Indicate true or false.

WR1-0 ◄——— 0 1 0 0 0 1 0 0
WR1-1 ◄——— 1 1 0 0 0 1 0 1

a. ___ Port A is an I/O port.

b. ___ Port A address is decremented after each byte is transferred.

c. ___ A timing byte follows the base register byte.

11-4 Refer to Question 11-3 above and indicate true or false.

a. ___ WR1-1 specifies that variable cycle length timing is used.

b. ___ The number of T-cycles used is 2.

c. ___ The $\overline{\text{IORQ}}$ signal is terminated ½ cycle early.

d. ___ The $\overline{WR}$ signal is terminated ½ cycle early.

e. ___ The $\overline{MREQ}$ signal is terminated ½ cycle early.

f. ___ The $\overline{RD}$ signal is terminated ½ cycle early.

11-5 Answer the following questions (yes or no) for the WR2 group whose base register is loaded with the following byte.

WR2-0 ◄——— 0 0 1 1 1 0 0 0

a. ___ Is the variable timing byte loaded after WR2-0?

b. ___ Will the port address be incremented after each byte is transferred?

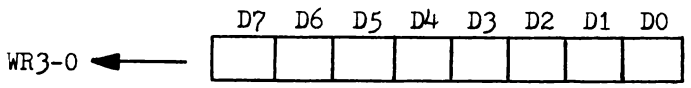
c. ___ Will the port address be decremented after each byte is transferred?

d. ___ Is the Port B address fixed?

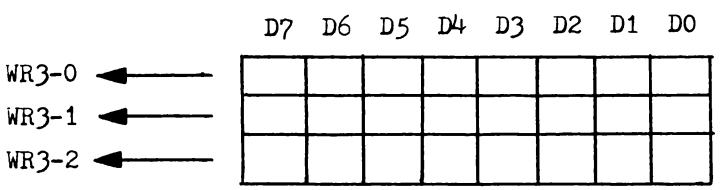
e. ___ Is the Port B an I/O port?

11-6 The WR3 group is concerned with the Mask and Match bytes. Construct the WR3-0 byte for the following conditions:

a. A Match byte follows.
b. A Mask byte follows:
c. Stop on Match.
d. Disable Interrupt.



11-7 The WR3 group of the Z-80/DMA is to be programmed such that the Match byte consists of all ones. However, the first and the last bits of this byte are to be ignored. When a match is obtained, the search must be stopped but an interrupt is not generated. Fill in the bits for the three bytes in the chart below.



11-8 The following statements relate to writing the WR4 group of bytes. The DMA is operated in the Continuous Mode and no interrupt control byte follows. The HEX code for the Port B Starting Address is $3B8D_H$. Indicate true or false.

a. ___ Bit D0 and D1 of WR4-0 are 1 and 0 respectively.

b. ___ Bits D6 and D5 of WR4-0 are 1 and 0 respectively.

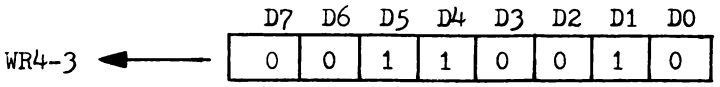
c. ___ Bit D4 of WR4-0 is a 1.

d. ___ WR4-2 contains 0 0 1 1 1 0 1 1 .

e. ___ WR4-1 contains 1 0 0 0 1 1 0 1 .

f. ___ Bit D7 of WR4-0 is a 0.

11-9 The Interrupt Control Register, WR4-3, contains the following byte.

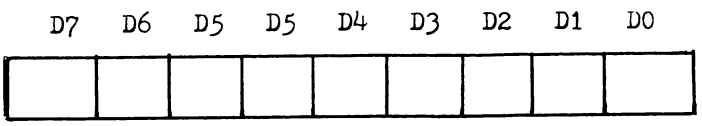


In this case the D2 and D1 bits in WR4-5 will be respectively:

a. 1 1
b. 0 0
c. 1 0
d. 0 1

11-10 Construct the byte for the Base Register of group WR5 for the following conditions:

a. Auto Restart on End-of-Block
b. Ready Active Low
c. CE/WAIT line signal is wait when the bus acknowledge signal input is active.



11-11 The following statements apply to the Base Register of the WR6 group. Indicate true or false for the command which results in the read-out from the Status Register.

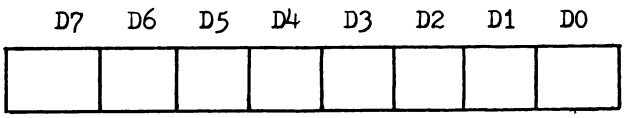
a. ___ The HEX code for the command is $B8_H$.

b. ___ The f4 bit of the command is 0.

c. ___ The f1 bit of the command is a 1.

d. ___ The f0, f2 and f3 bits of the command are all ones.

11-12 The HEX code BB is used in the WR6-0 register. Construct the byte for the WR6-1 register which will enable us to read out the low byte of Port A address.



11-13 The statements below refer to the DMA Status Register. Indicate true or false.

a. ___ A 1 in D1 indicates Ready Active status.

b. ___ The End-of-Block condition is indicated by a 0 in bit position D5.

c. ___ Bit position D2 must always be a 0.

d. ___ A Match found condition is initiated by a 0 in bit D4 position.

12 Z-80 SERIAL INPUT OUTPUT CHIP

CHAPTER OBJECTIVES

1. To introduce students to the programmable serial I/O chip.
2. To present the principal features of the Z-80 serial input-output (SIO) chip.
3. To present and describe the generalized architecture of the SIO Receive Data path for both the asynchronous and the synchronous modes of operation.
4. To describe byte-oriented, synchronous, protocols namely, the monosync, bisync and external methods of character synchronization.
5. To describe the operation and use of the Cyclic Redundancy Check (CRC) character.
6. To examine the Bit-oriented protocol (SDLC Mode).
7. To present and describe the generalized architecture and operation of the SIO Transmit Data path for both the asynchronous and the synchronous modes of operation.
8. To present the package pin assignments and functions in convenient tabular form.
9. To discuss the SIO timing involved in the read, the write, the interrupt acknowledge and the return from interrupt cycles.

TEXTBOOK REFERENCES

For reviewing material in the main hardcover textbook, relevant to the topics in this chapter, the following chapters and/or sections are suggested:

1. For general discussion of Serial I/O transfers - Chapter 11
2. For general discussion of programmable I/O transfers - Chapter 12
3. For conversion and synchronization problems in Serial I/O transfers - Sec. 11-2

4. For description of the UART - Sec. 12-2

5. For multiple buffering of data - Sec. 12-2-1.4

6. For UART Error indications - Sec. 12-2-1.5

7. For UART initialization - Sec. 12-2-1.6

8. For general UART operation and block diagram - Sec. 12-2-1.8

12-1 INTRODUCTION

The Serial Input-Output (SIO) chip provides programmable, full duplex, two independent channels which have separate control and status lines for interfacing the Z-80/CPU with modems and other devices which require serial digital communication capabilities. Basically, the SIO functions as a serial-to-parallel and parallel-to-serial converter and controller. Under program control, the SIO can be reconfigured to operate under several different conditions and modes so that it can be interfaced with I/O devices having different word lengths and a wide range of operating speeds.

The SIO handles several protocols. They include both synchronous and asynchronous protocols as well as the bit and byte oriented functions, usually performed by UARTS and USARTS. All incoming and outgoing data are fully buffered. The Receiver data are quadruple buffered and the transmitter data are double buffered. In the Z-80 μC system, the SIO can be interfaced with either the CPU or the DMA chip. Also, it can be used in the interrupt daisy chain and has the capability of providing interrupt vectors without any additional logic.

12-2 PRINCIPLE FEATURES OF THE SIO

1. The SIO is a third generation chip which uses the N-channel, depletion-mode, silicon-gate technology.

2. A single, +5 $\pm$5% volts power supply is required.

3. A single-phase, 5-volt clock is used.

4. All inputs and outputs are fully TTL compatible.

5. The chip operates in a temperature environment of $0^{o}C$ to $+70^{o}C$.

6. In the synchronous operating mode, the following data rates can be used:

   a. With 2.5 MHz system clock - 0-550 K bits/second
   b. With 4.0 MHz system clock - 0-880 K bits/second

7. The SIO has the capability to operate with 5, 6, 7, or 8 bits per character and

odd or even parity or no parity at all.

8. Parity, framing and overrun errors can be detected and the SIO can operate in four clock modes, x1, x16, x32, and x64.

12-3 THE SIO ARCHITECTURE

12-3.1 THE Generalized SIO Block Diagram

1. Figure 12-1 shows the generalized block diagram of the SIO. On the CPU side, the chip uses the 8-bit data bus for transfer of data between the CPU and the SIO as well as the transfer of command words from the CPU to the SIO. Control signals are sent to the SIO on a 6-bit control bus.

2. The Internal Control Logic and the Interrupt Control Logic blocks essentially perform the same functions that were previously described for the PIO chip in Chapter 6 and the CTC chip in Chapter 8.

3. Each channel has its own group of read/write registers which include five 8-bit control registers, two sync-character registers and two status registers. Each channel also has discrete control logic and status logic which makes it possible for the SIO to communicate with Modems or other I/O devices.

4. For both channels, the write registers are designated WR0 - WR7 and the read registers are designated RR0 - RR2. The function performed by each register is given below:

   Write Registers

   WR0 - Contains Register Pointers and initialization commands for the different modes, including CRC initialization.

   WR1 - Defines receive/transmit interrupts and the data transfer mode.

   WR2 - Contains the Channel B interrupt vector (not channel A).

   WR3 - Contains Receiver parameters and controls.

   WR4 - Contains various miscellaneous parameters for different Receive/Transmit modes.

   WR5 - Contains parameters and controls for the transmit operation.

   WR6 - Contains the sync character or the SDLC (Synchronous Data Link Control) address field.

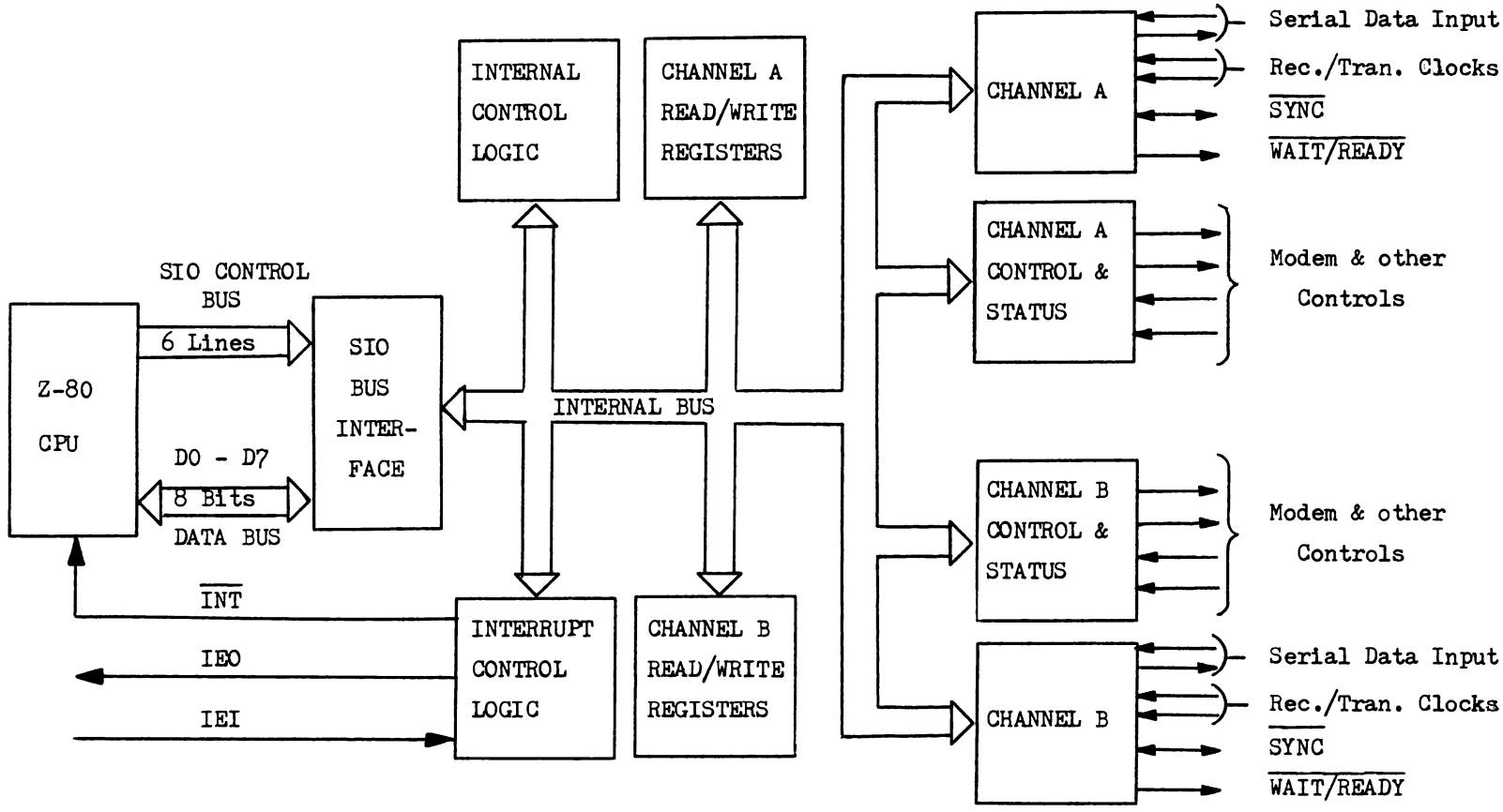


Fig. 12-1 Generalized SIO block diagram.

WR7 - Contains the sync character or the SDLC flag.

Read Registers

RR0 - Holds the buffer status, interrupt status and external status words for the Receive/Transmit operations.

RR1 - Contains the status for the Special Receive Condition.

RR2 - Contains the Channel B Modified Interrupt Vector (Not Channel A).

5. In both channels the associated logic formats, synchronize and validates the data transferred in and out of that channel.

6. The Modem Control signals, for both channels, are general-purpose and can be used for functions other than controlling the Modems.

7. The interrupt control logic provides the automatic interrupt vectoring capabilities by determining which channel, and which I/O device within that channel, has the highest priority.

8. In the SIO, Channel A has a higher priority over Channel B. Also, the Receive Status, the Transmit and the External Status interrupts are assigned priorities, in that order, within each channel.

12-3.2 The Receive Data Path

1. Figure 12-2 is a simplified block diagram of the path taken by the incoming data in the SIO.

2. The incoming data is a serial bit stream in which the word length could be from 5 to 8 bits. Also, the serial data could be transmitted in either the synchronous or the asynchronous mode. (See Sec. 11-3.3 in the main hardcover textbook for a brief description of these modes). We will follow the data path in Fig. 12-2 for both these data transmission modes.

12-3.2.1 The Asynchronous Mode

1. The serial data stream enters the SIO chip via the RxDA line and goes through a 1-bit time delay.

2. From this point, the data bit stream can proceed in one of the two possible paths depending upon the word length. If the character is 5 or 6 bits long, it enters the 3-bit Buffer and is then loaded into the 8-bit Receive Shift Register.

3. If the incoming character is 7 or 8 bits long, it bypasses the 3-bit Buffer and

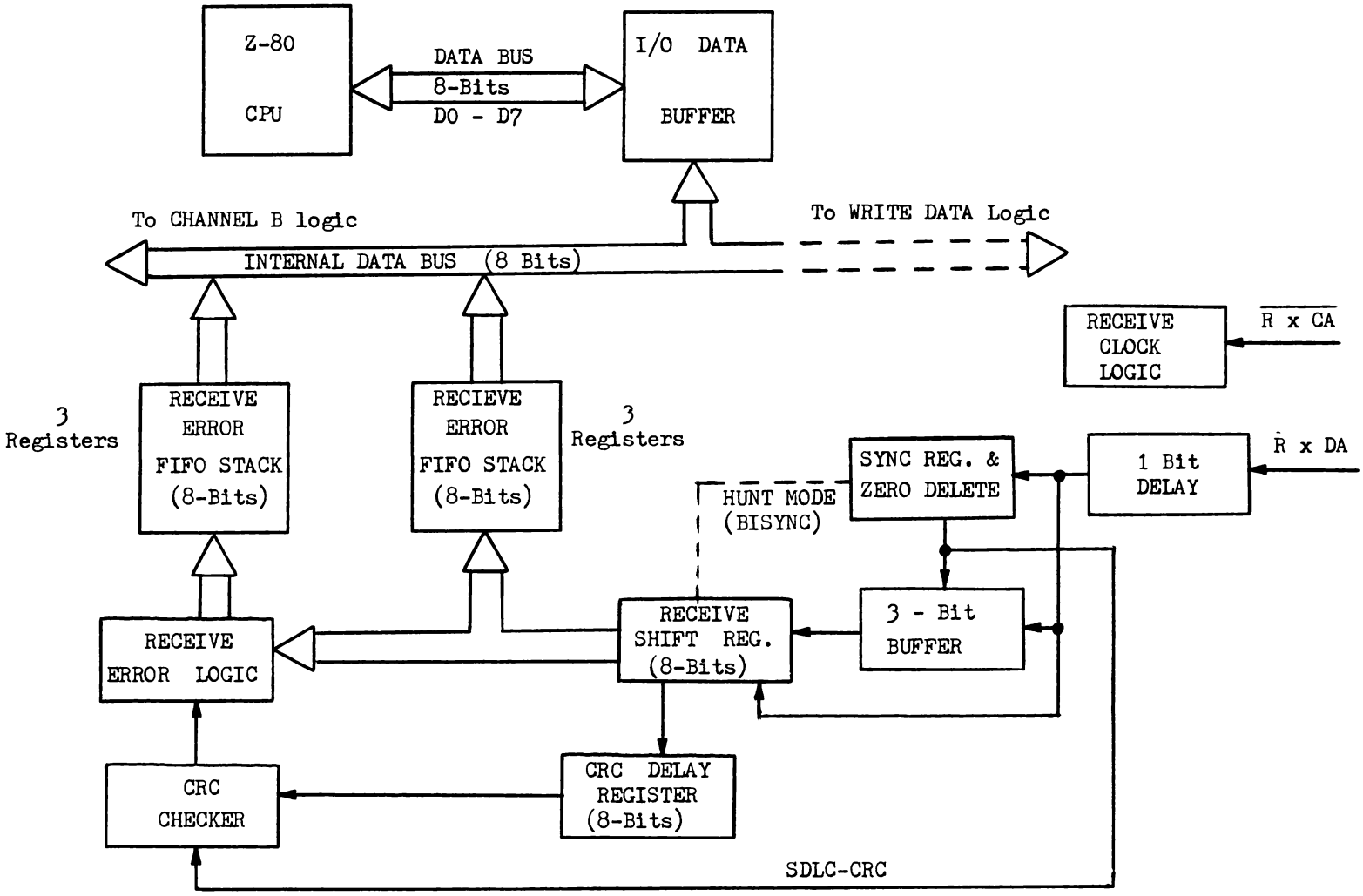


Fig. 12-2 Block diagram of receive data path.

is directly loaded into the 8-bit Receive Shift Register.

4. The reason for the above two procedures is that the Receive Shift Register is an 8-bit serial-in/parallel-out shift register. The output of this register is always an 8-bit parallel word regardless of the number of bits in the incoming word. To accomplish this goal, the SIO is designed to operate as follows:

   a. The receiver logic automatically inserts 1s when a character length of less than 8 bits is used in the incoming serial bit stream.

   b. If the parity is enabled, and the incoming word length is 8 bits, then the parity bit is stripped from the word and only the 8 data bits are converted into a parallel format.

   c. If the incoming word is 7 bits long plus parity, then the parity is not stripped but carried along.

   d. If the incoming word is 5 or 6 bits long, then the 3-bit Buffer inserts 1, 2 or 3 bits (binary 1s) depending upon whether the word is 5 or 6 bits and whether a parity bit is included or not.

5. The artificially inserted 1 bits are always in the higher significant bit positions of the parallel formatted word. For example, if the incoming word were 5 bits long and had a parity bit, then the assembled word in the Receive Shift Register would look as shown in Fig. 12-3.

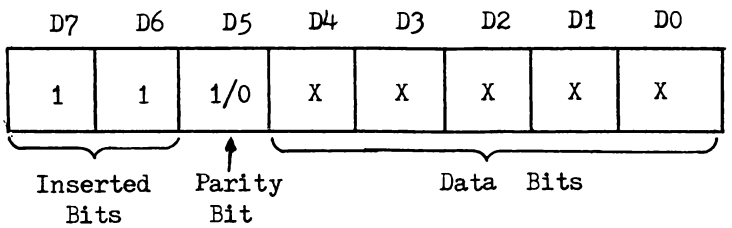


Fig. 12-3 Artificially inserted bits in serial bit stream.

6. The 8-bit word is then sent to the 3-register Receiver Data Stack which operates on a First-In, First-Out (FIFO) basis. From this stack the 8-bit word is transferred to the Z-80/CPU via the Internal Data Bus, the I/O Data Buffer and the CPU Data Bus (D0 - D7).

7. Notice that the 8-bit parallel data word is quadruple buffered since it passes through the 3 FIFO registers and the Receive Shift Register. This time delay is provided so that the CPU has ample time to service any interrupts that may come up while the SIO keeps accepting and formatting the incoming data words.

8. The formatted 8-bit word is also sent to the Receive Error Logic. This logic

checks for parity, framing and overrun errors and generates error status word. This word is then sent to the 3-register, 8-bit, Receive Error Stack. This is also a FIFO stack.

9. Thus, each data word is quadruple buffered and its corresponding error status word is likewise quadruple buffered, and for the same reason. Since both stacks read out to the Internal Data Bus, correlation between the two is maintained by first reading out the error status word and then the corresponding data word. A special interrupt vector is generated if an error is indicated by the status word.

12-3.2.2 The Synchronous Mode

A. Byte-Oriented Protocols

The synchronous mode is a little more complex than the asynchronous mode. In the SIO modes of data transmissions we use some terms which are explained below:

1. Monosync Synchronization of Character - This is the situation where the actual character or data word is preceded by a preset 8-bit SYNC pattern which is inserted by the transmitter. The SIO has the same pattern of bits prestored in register WR7. When a match between the incoming data stream and this pattern in WR7 is made, synchronization is achieved.

2. Bisync Synchronization of Character - This is very similar to Monosync synchronization except that the same SYNC bit pattern is prestored in registers WR6 and WR7. This essentially implies a 16-bit SYNC pattern. Synchronization is achieved only when the desired data character is preceded by two contiguous SYNC bit patterns. The data character is captured and synchronized only when a match is achieved between the 2 SYNC patterns and the contents of WR6 and WR7. Note that in both the Monosync and the Bisync modes, the pin labelled $\overline{\text{SYNC}}$ of the SIO chip is used as an output. It is active during the part of the receiver clock that detects the SYNC character or pattern of bits.

3. External Synchronization - Character synchronization is achieved externally in this mode. The $\overline{\text{SYNC}}$ pin is used as an input that indicates that synchronization is achieved. After the sync pattern is detected by comparing the incoming bit stream with the contents of WR7, the logic in the I/O must wait for two cycles of the Receive Clock before the $\overline{\text{SYNC}}$ input line is activated. The assembly of the data character begins on the positive-going edge of the RxC receive clock that precedes the falling edge of the $\overline{\text{SYNC}}$ signal.

4. The Hunt Phase - In all three above-mentioned modes of character synchronization, the SIO first goes through a hunt phase during which it looks for character synchronization. The SIO goes into the hunt phase after it is reset and and only if the receiver is enabled. The data transfer begins only when proper synchronization is achieved.

5. Cyclic Redundancy Check (CRC) Character - This is a 16-bit error checking code that is appended to each record that is transmitted and subsequently received. A record is formed by several characters of groups of words. The CRC is generated by counting the total number of bits (both is and 0s) involved in the record and a mathematical formula in the transmitter. In the receiver the bits in the receiver are again checked using the CRC code appended to the record.

The message formats for the Monosync, Bisync and the External sync modes are shown in Fig. 12-4.

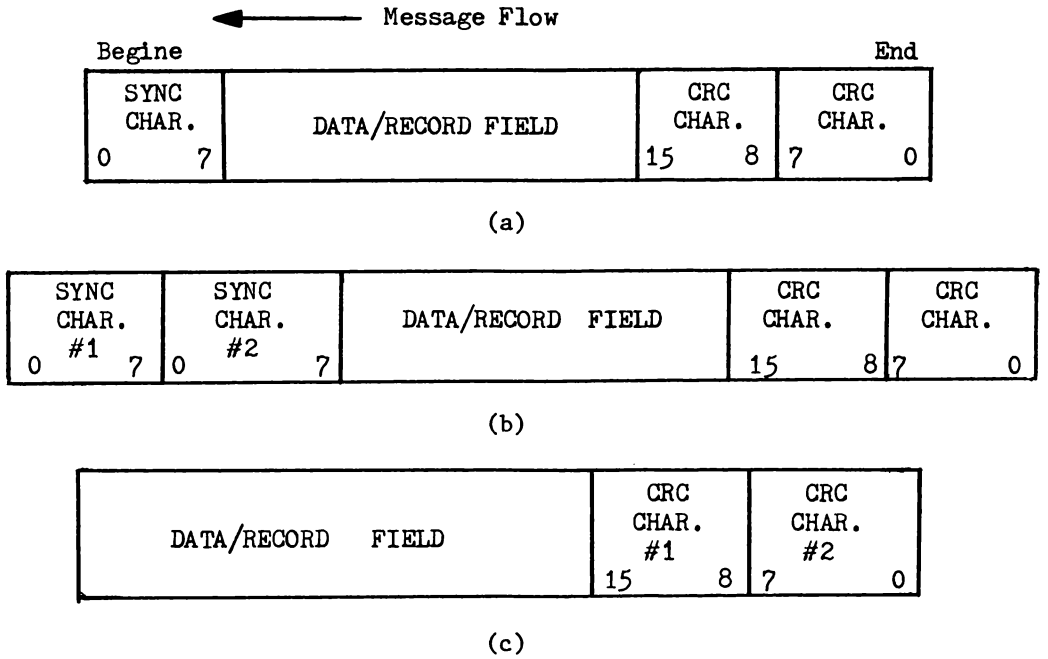


Fig. 12-4 Synchronous message transmission formats. (a) Monosync, internal sync detect, format. (b) Bisync internal sync detect, format. (c) External sync detect format.

6. The Receive Operation - To understand the receive operation in the synchronous mode, refer to Fig. 12-2. The path taken by the incoming data through the receiver depends upon the character length and the transmission mode used.

   a. The receive operation begins with the hunt phase. The incoming data bit stream is monitored for the SYNC pattern bits. If the SIO is programmed for the Monosync mode, a match between the contents of WR7 and the incoming bits results in synchronization and the succeeding words are then captured and handled in exactly the same manner as previously described

for the asynchronous mode. In this mode the comparison is made with the incoming bits assembled in the Sync Register and the WR7.

b. In the Bisync mode, the first 8 bits of the incoming data stream are checked in the SYNC Register and then transferred into the Receive Shift Register. The next 8 bits are then shifted into the Sync Register. Synchronization is established when the contents of the Receive/Sync Register are matched with the contents of the WR6/WR7 register. The data/record characters are then handled as previously explained.

B. Bit Oriented Protocol (SDLC Mode)

1. In addition to the Monosync and the Bisync synchronous modes which are byte-oriented protocols, the SIO is also designed to handle a bit-oriented protocol, called the IBM SDLC (Synchronous Data Link Control).

2. Operation in this mode also starts with the hunt phase. The incoming data bit stream first goes through the Receive SYNC Register. When five contiguous 1 bits are received, the sixth bit is monitored. If it is a 0 it is deleted from the bit stream. If it is a 1, then the seventh bit is checked. If the seventh bit is a 0, then it is interpreted that a Flag sequence has been received. If the seventh bit is a 1, then that is interpreted that an Abort Sequence has been received.

3. The reformatted word is then transferred to the 3-bit register and from there to the Receive Shift Register. The contents of this register are then compared with the prestored contents of the WR7 register, called the flag register. Once the flag character match is established, subsequent data/record are transmitted and handled as explained before.

4. The CRC characters are also appended to the data/record field in the SDLC mode operation.

5. In both the synchronous and the SDLC modes, the CRC character is checked for errors, and so is passed through the CRC Checker (See Fig. 12-2.) However, in the Bisync mode the SYNC-CRC is passed through an 8-bit delay register to allow the CPU ample time to make the necessary decision and to include the data character in the CRC. The SDLC-CRC is not delayed because the SIO contains logic, in this mode, that determines the bytes by means of which the CRC is checked.

12.3.3 The Transmit Data Path

1. Figure 12-5 is a simplified block diagram of the path taken by the serial data going out from the SIO.

2. The 8-bit parallel output data word from the Z-80/CPU comes to the SIO on the Do - D7 Data Bus and is then loaded into the I/O Data Buffer.

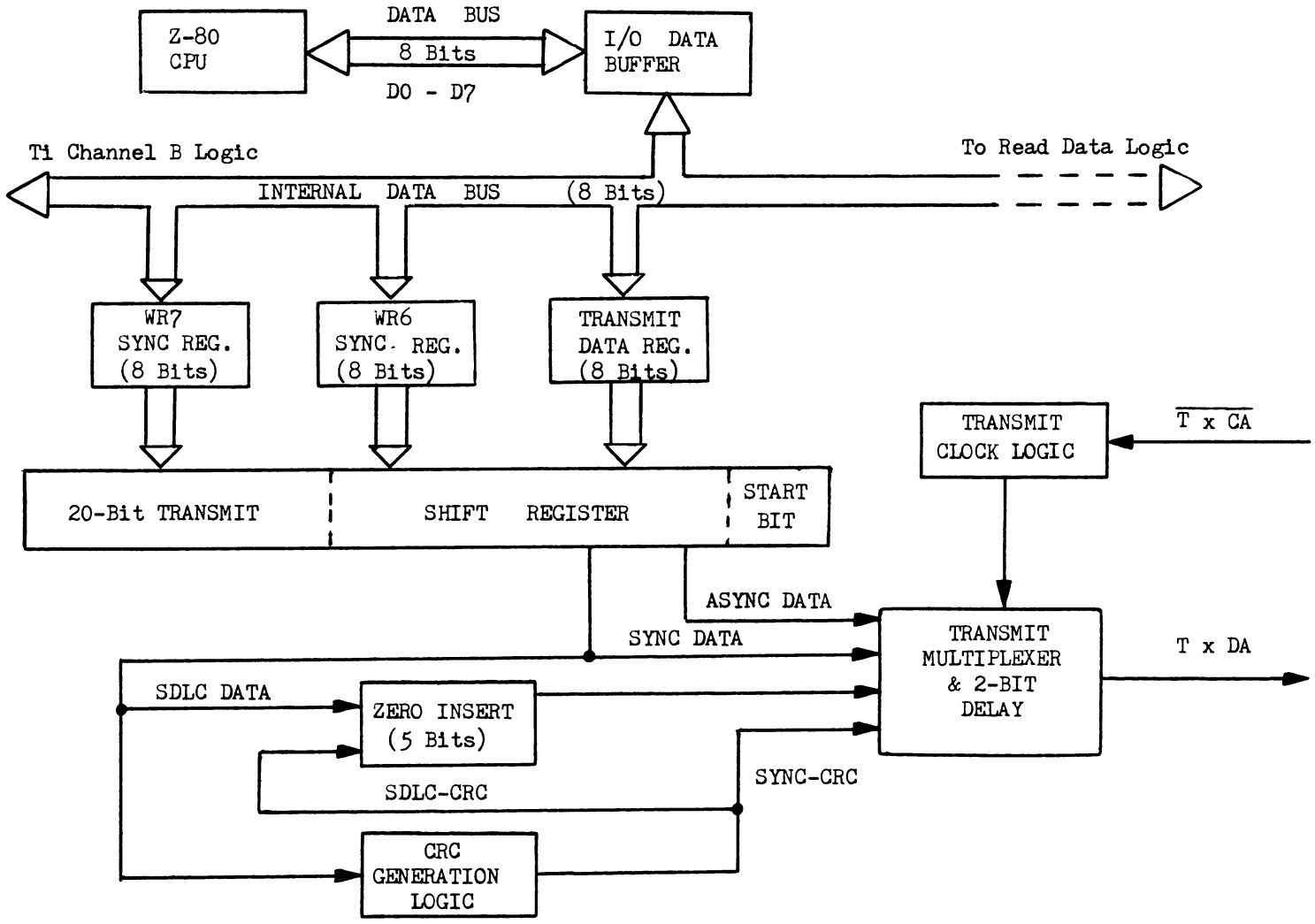


Fig. 12-5 Block diagram of transmit data path.

3. From there it is loaded into the 8-bit Transmit Data Register via the Internal Data Bus. The data word is then transmitted out of the SIO as explained below, depending upon the transmission mode used.

12-3.3.1 The Asynchronous Mode

1. In this mode, the serial data words are shifted out at the TxDA terminal at a clock rate equal to 1, 1/16, 1/32 or 1/64 of the clock rate inputted to the Transmit Clock Logic at the TxCA terminal.

2. The data word is automatically formatted by the logic in the SIO which adds the start bit, the parity bit (odd, even or no parity bit depending upon how it is programmed) and the appropriate number of stop bits, also depending upon the program.

3. If the character word to be transmitted is 6 or 7 bits long, the SIO automatically ignores the unused bits.

4. If the character word to be transmitted is 5 bits or less, then the transmitted word is formatted according to a pre-established convention which is described in the programming section of the ZILOG Z-80/SIO Technical Manual © 1977 in tabular form for the WR5 register format.

12-3.3.2 The Synchronous Mode

1. The Transmit Multiplexer receives its input from four possible data paths. It then formats the word appropriately and transmits it out serially in the T x DA output terminal.

2. WR6 and WR7 contain the SYNC characters which are transferred into the 20-bit Shift Register at the beginning of the message. They are also used as time fillers in the middle of a message if a Transmit Underrun condition happens.

3. The SYNC pattern word(s) is/are followed by the actual data word or character. The two CRC characters are then generated and appended as shown in Fig. 12-4.

4. As mentioned before, the SDLC message is called the frame whose format is shown in Fig. 12-6.

5. The SDLC message opens and closes with the 8-bit flag which is 01111110. It also contains an 8-bit address field. This is called the secondary station address. It is used in the address search mode which utilizes this address to either accept or reject the frame.

6. From the 20-bit Transmit Shift Register, the SDLC data word is shifted into the Zero Insert Logic block. When the closing and opening flags are transmitted,

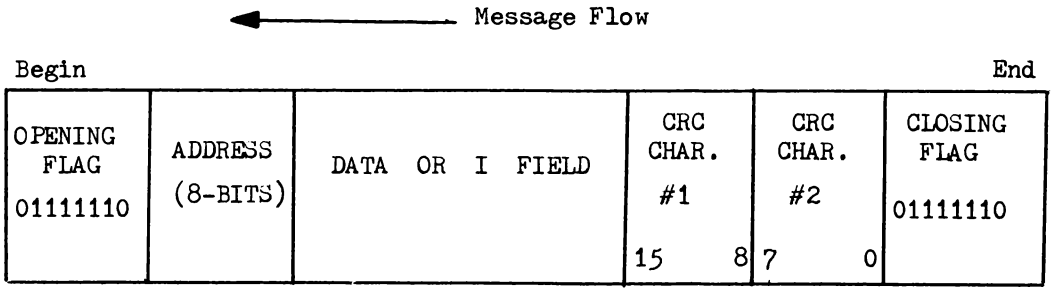


Fig. 12-6 The SDLC Message Format.

6. this block is disabled. But, when the other fields are transmitted, a 0 is inserted after 5 contiguous 1s in the data stream.

7. In the SDLC mode, the data generated by the CRC Generator Logic is also routed through the Zero Insert Logic Block.

12-4 PACKAGE PIN ASSIGNMENTS AND FUNCTIONS

The SIO is packaged in a standard 40-pin DIP package. The various pins and their respective functions, along with their designations, are presented in convenient tabular form. Functionally, the pins can be divided into 9 groups. Because of package pin limitations, a peculiar problem arises on the SIO on the I/O side of the chip. Three lead bonding configurations are possible. The group on the CPU side remains constant and is shown in Fig. 12-7. The three configurations on the I/O side will be covered after we have presented the various pin functions first.

12-4.1 The CPU side Pins

1. Figure 12-7 shows the 5 groups of pins on the CPU side of the SIO chip. These pins and their groupings remain constant regardless of the three different configurations on the channel side of the chip.

2. The pin functions and their designations are shown in Tables 12-1 through 12-5. Some of the lines in these tables may indicate signal flow in one direction and others in the other direction. The function tables clarify these with the following arrow symbols:

   SIO ——→ Indicates signals flowing from the SIO.

   SIO ←—— Indicates signals flowing into the SIO.

   SIO ←——→ Indicates bidirectional flow on the same line(s).

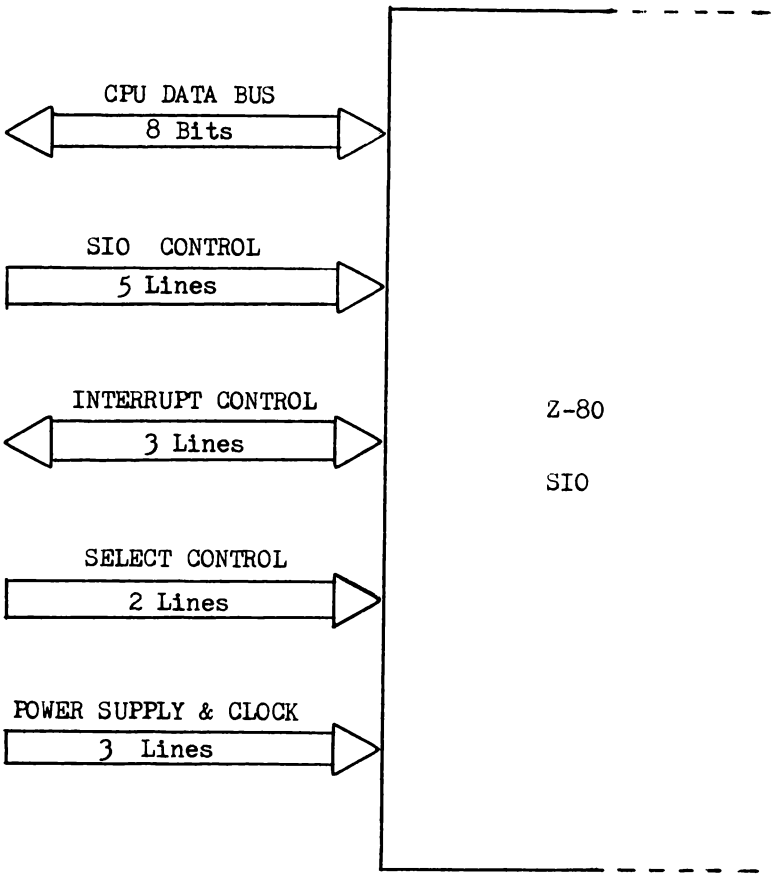


Fig. 12-7 Block diagram of the SIO package pin groups on the CPU side.

Table 12-1 Data transfer bus group.

PIN DESIGNATIONS	SIGNAL FLOW DIRECTIONS	FUNCTIONS
D0 - D7	SIO ◄——►	8-bit, tristate, CPU data bus, is active high. It is used for data and commands transfers between the SIO and the CPU.

Table 12-2 SIO Control group

PIN DESIGNATIONS	SIGNAL FLOW DIRECTIONS	FUNCTIONS
$\overline{CE}$	SIO ◄———	Chip enable, active low signal that commands the SIO chip to transmit data during the read cycle. During the write cycle, the SIO accepts data or command words from the CPU.
$\overline{RESET}$	SIO ◄———	Reset signal, active low. It does the following: 1) Disables transmitters & receivers, 2) Disables all interrupts, forces the modem controls to the high state, 4) Forces T x DA and T x DB marking. After the SIO is reset, the control registers must be rewritten before data is received or transmitted.
$\overline{M1}$	SIO ◄———	Machine Cycle One signal, active low. When $\overline{M1}$ and $\overline{RD}$ are simultaneously active the CPU is fetching an instruction from the memory. When M1 and $\overline{IORQ}$ are simultaneously active, the CPU is acknowledging an interrupt. The SIO then places its interrupt vector on the CPU data bus provided it has the highest priority and it has requested an interrupt.
$\overline{IORQ}$	SIO ◄———	I/O Request from CPU signal, active low, is used for transferring data between CPU and SIO as well as for sending commands to the SIO. The SIO write cycle is initiated by the active status of $\overline{CE}$ and $\overline{IORQ}$ and inactive status of $\overline{RD}$. During the write cycle the CPU writes either the data word or the control word into the selected channel as specified by the C/$\overline{D}$ signal. The SIO read cycle is initiated by simultaneous activation of the $\overline{CE}$, $\overline{IORQ}$ and $\overline{RD}$ signals. During the read cycle, the SIO channel selected by the B/$\overline{A}$ signal transfers the data word to the CPU via the data bus.
$\overline{RD}$	SIO ◄———	Read cycle status, active low, signal when used simultaneously in conjunction with $\overline{CE}$ and $\overline{IORQ}$, it signals the SIO to transfer data to the CPU. During the write cycle, $\overline{RD}$ is inactive while $\overline{CE}$ and $\overline{IORQ}$ is active.

Table 12-3 Interrupt Control group.

PIN DESIGNATIONS	SIGNAL FLOW DIRECTIONS	FUNCTIONS
$\overline{INT}$	SIO ◄——	Interrupt Request. The $\overline{INT}$ is pulled low by the SIO chip when it is requesting an interrupt.
IEI	SIO ——►	Interrupt Enable In, active high. Used in the system-wide priority interrupt daisy chain. A high or active signal indicates that no other device with higher priority is currently being serviced.
IEO	SIO ◄——	Interrupt Enable Out, active high. Also used in the interrupt priority daisy chain. It is active only when IEI is active and the CPU is not servicing an interrupt from the SIO. When inactive, it blocks interrupt signals from lower priority I/O devices while the CPU is servicing an I/O device having a higher priority.

Table 12-4 Select Control group.

PIN DESIGNATIONS	SIGNAL FLOW DIRECTIONS	FUNCTIONS
C/$\overline{D}$	SIO ◄——	Control or Data Select Signal. In the high state it informs the SIO that the word outputted by the CPU on the data bus must be accepted by the selected channel as a command or a control word. When in the low state, the selected channel will interpret and accept the word on the data bus as a data word.
B/$\overline{A}$	SIO ◄——	Channel Select Signal. When high, Channel B is selected and when low Channel A is accessed. Transfers are then made between the CPU and the selected channel. This signal is often transmitted to the SIO on the A0 line of the address bus from the CPU.

Table 12-5 Power Supply and Clock group.

PIN DESIGNATIONS	SIGNAL FLOW DIRECTIONS	FUNCTIONS
ø	SIO ◄——	Single-phase, TTL-level, clock input.
+5V	SIO ◄——	This is the single power supply of +5 ± 5% volts that is required.
GND	SIO ◄——	This is the power supply ground.

Table 12-6 CPU side Pin Number assignments.

PIN DESIGNATIONS	PIN NUMBERS	PIN DESIGNATIONS	PIN NUMBERS
CPU Data Bus		Interrupt Control	
D0	40	$\overline{INT}$	5
D1	1	IEI	6
D2	39	IEO	7
D3	2		
D4	38		
D5	3	Select Control	
D6	37	C/$\overline{D}$	33
D7	4	B/$\overline{A}$	34
SIO Control		Power Supply & Clock	
$\overline{CE}$	35	ø	20
$\overline{RESET}$	21	+5V	9
$\overline{M1}$	8	GND	31
$\overline{IORQ}$	36		
$\overline{RD}$	32		

12-4.2 The I/O Side Pins

1. From the presentation in Section 12-4.1 and Table 12-6 we find that 21 of the 40 package pins have been used by the CPU side, leaving 19 pins for the I/O side.

2. The demands of the I/O functions are such that these 19 pins are just not adequate for all I/O situations. Some compromises have to be made. In the SIO, these compromises are made such that in one case two signals from the chip are internally bonded together to one package pin. In two other instances one output from the chip is not brought out to a package pin at all. We will look at each of these three output pin configurations after we have discussed all the functions on the I/O side of the chip.

3. Figure 12-8 is the block diagram of the SIO package pin groups on the I/O side. Notice that the number of lines for Channel A are fixed. The above-mentioned pin limitation compromises are made only in Channel B and that is why alternate numbers are given for the lines involved.

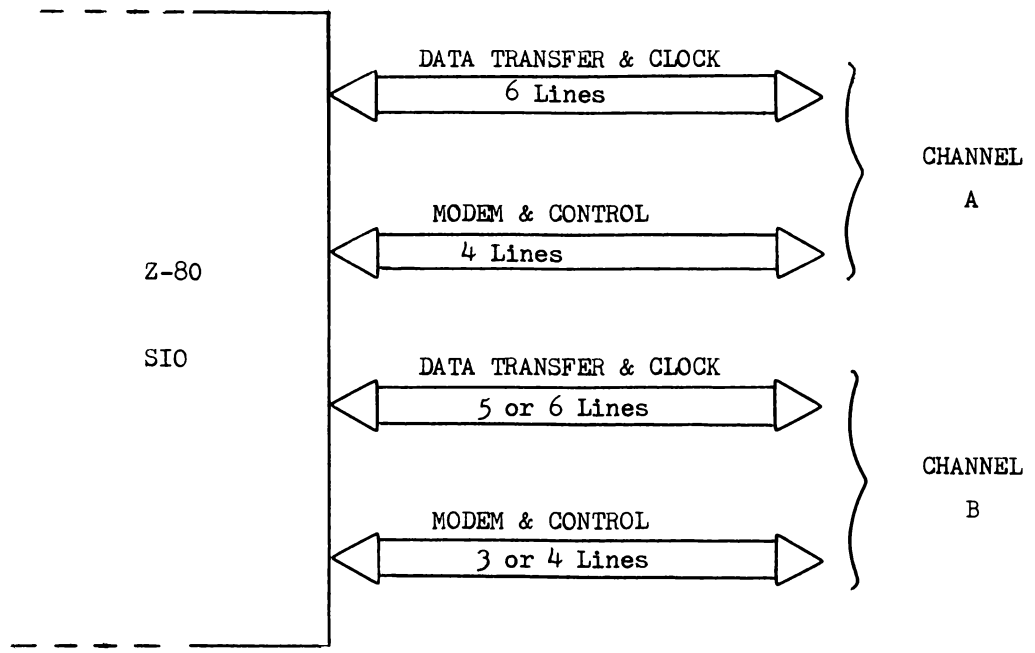


Fig. 12-8 Block diagram of the SIO package pin groups on the I/O side.

Table 12-7 Data transfer and clock group.

PIN DESIGNATIONS	SIGNAL FLOW DIRECTIONS	FUNCTIONS
RxDA, RxDB	SIO ◄———	Receive Data, active high. Serial data, TTL compatible, signals are inputted to the SIO.
$\overline{\text{RxCA}}$, $\overline{\text{RxCB}}$	SIO ◄———	Receive Clocks, active low. These clock inputs may be driven by the Z-80/CTC timer, discussed in chapter 8, for the programmable generation of the baud rate. In asynchronous mode of operation, the receiver clocks could be 1, 16, 32 or 64 times the incoming data rate. Incoming data are sampled on the positive-going edge of the $\overline{\text{RxC}}$ signal. Both inputs are buffered by Schmitt Triggers.
TxDA, TxDB	SIO ———►	Transmit Data, active high. Serial data, TTL compatible, signals are outputted by the SIO.
$\overline{\text{TxCA}}$, $\overline{\text{TxCB}}$	SIO ◄———	Transmitter clocks, active low. These clock inputs may be driven by the Z-80/CTC timer, discussed in chapter 8, for the programmable generation of the baud rate. In asynchronous mode of operation, the transmitter clocks could be 1, 16, 32 or 64 times the outgoing data rate, but the clock multiple factor for the receiver and the transmitter must be the same. The transmitted data (TxD) changes states at the negative-goingedge of the TxC pulse. Both inputs are buffered by Schmitt Triggers.
$\overline{\text{SYNCA}}$, $\overline{\text{SYNCB}}$	SIO ◄———►	Synchronization, bidirectional signals, active low. In the asynchronous mode, change of signals (high-to-low or low-to-high) affect the SYNC/HUNT status bits in RRO. When externally synchronized, $\overline{\text{SYNC}}$ must be driven low on the second positive-going edge of the receiver clock ($\overline{\text{RxC}}$) after the first rising edge of ($\overline{\text{RxC}}$) on which the last character of the SYNC character was received. This means that after the sync pattern is detected, the $\overline{\text{SYNC}}$ input must be activated only after waiting for 2 full $\overline{\text{RC}}$ cycles. When used in the internal synchronization mode (Bisync or Monosync), these pins are active as outputs during the portion of the $\overline{\text{RxC}}$ cycle in which SYNC characters are recognized.

Table 12-7 (Continued) Data transfer and clock group.

PIN DESIGNATIONS	SIGNAL FLOW DIRECTIONS	FUNCTIONS
$\overline{W/RDYA}$, $\overline{W/RDYB}$	SIO ⟶	Wait/Ready Signal lines. The lines are open (drain open) when they are programmed for the WAIT function. They can be either in the high or the low state when programmed for the READY function. As a WAIT function they are used to synchronize the CPU to the SIO data transfer rate. When used with a DMA controller, they can be used for the READY function.

Table 12-8 Modem and other control group.

PIN DESIGNATIONS	SIGNAL FLOW DIRECTIONS	FUNCTIONS
$\overline{RTSA}$, $\overline{RTSB}$	SIO ⟶	Ready to Send, active low, signals. This signal goes low when the RTS bit is set. In the asynchronous mode, when the RTS bit is reset, the output line goes high after the transmitter is empty. When used in the synchronous mode, the $\overline{RTS}$ lines only follow the state of the RTS bit.
$\overline{CTSA}$, $\overline{CTSB}$	SIO ⟵	Clear to Send, active low, signals. If programmed as Auto Enables, low signal inputs on these lines enables the corresponding transmitter. Otherwise, they can be programmed and used as general-purpose inputs. Logic level transitions in either direction on these lines are detected by the SIO which then sends an interrupt request to the CPU. Both lines are buffered with Schmitt Triggers.
$\overline{DTRA}$, $\overline{DTRB}$	SIO ⟶	Data Terminal Ready, active low. These lines follow the DTR bit state as determined by the program. They can also be used as general-purpose outputs.

Table 12-8 (Continued) Moden and other control group.

PIN DESIGNATIONS	SIGNAL FLOW DIRECTIONS	FUNCTIONS
$\overline{DCDA}$, $\overline{DCDB}$	SIO ◄———	Data Carrier Detect, active low. If the SIO is programmed for Auto Enables, these lines function as receive enables. Otherwise, they can be used as general-purpose inputs. Logic level transitions in either direction in these lines are detected by the SIO which then sends an interrupt request to the CPU. Both lines are buffered with Schmitt Triggers.

4. The package pin assignments for Channel A remain the same, as shown in Table 12-9, regardless of the three configurations for Channel B.

Table 12-9 Channel A Pin Number assignments.

PIN DESIGNATIONS	PIN NUMBERS	PIN DESIGNATIONS	PIN NUMBERS
Data Transfer & Control		Modem & Control	
RxDA	12	$\overline{RTSA}$	17
$\overline{RxCA}$	13	$\overline{CTSA}$	18
TxDA	15	$\overline{DTRA}$	16
$\overline{TxCA}$	14	$\overline{DCDA}$	19
$\overline{SYNCA}$	11		
$\overline{W/RDYA}$	10		

5. Because of the previously-mentioned limitations imposed by the availability of only 40 pins on the package, some sacrifices are made in Channel B signals, namely Receive Clock, Transmit Clock, Data Terminal Ready and the SYNC signals. Two of the above four signals are internally bonded to a single package pin in some cases.

6. In the SIO/0 configuration, all the 4 above-mentioned signals are available but

$\overline{\text{TxCB}}$ and $\overline{\text{RxCB}}$ are internally bonded together. The resulting pin number assignment is shown in Table 12-10.

Table 12-10 The SIO/0, Channel B pin number assignments.

PIN DESIGNATIONS	PIN NUMBERS	PIN DESIGNATIONS	PIN NUMBERS
Data Transfer & Clock		Modem & Control	
RxDB	28	$\overline{\text{RTSB}}$	24
$\overline{\text{RxCB}}$, $\overline{\text{TxCB}}$	27	$\overline{\text{CTSB}}$	23
TxDB	26	$\overline{\text{DTRB}}$	25
$\overline{\text{SYNCB}}$	29	$\overline{\text{DCDB}}$	22
$\overline{\text{W/RDYB}}$	30		

7. In the SIO/1 configuration, the DTRB signal is sacrificed and not connected. However, the $\overline{\text{TxCB}}$, the $\overline{\text{RxCB}}$ and the $\overline{\text{SYNCB}}$ signals are available. The pin number assignments are shown in Table 12-11.

Table 12-11 The SIO/1, Channel B pin number assignments.

PIN DESIGNATIONS	PIN NUMBERS	PIN DESIGNATIONS	PIN NUMBERS
Data Transfer & Clock		Modem & Control	
RxDB	28	$\overline{\text{RTSB}}$	24
$\overline{\text{RxCB}}$	27	$\overline{\text{CTSB}}$	23
TxDB	25	$\overline{\text{DCDB}}$	22
$\overline{\text{TxCB}}$	26		
$\overline{\text{SYNCB}}$	29		
$\overline{\text{W/RDYB}}$	30		

8. In the SIO/2 configuration, the $\overline{SYNC}$ is sacrificed and not connected. However, the T x CB, the R x CB and the $\overline{DTRB}$ signals are available. The pin number assignments are shown in Table 12-12.

Table 12-12 The SIO/2, Channel B pin number assignments

PIN DESIGNATIONS	PIN NUMBERS	PIN DESIGNATIONS	PIN NUMBERS
Data Transfer & Clock		Modem & Control	
R x DB	29	$\overline{RTSB}$	24
$\overline{R \times CB}$	28	$\overline{CTSB}$	23
T x DB	26	$\overline{DTRB}$	25
$\overline{T \times CB}$	27	$\overline{DCDB}$	22
$\overline{W/RDYB}$	30		

12-5 THE SIO TIMING

In this section we briefly describe and discuss the SIO timing charts involved in the Read, Write, Interrupt Acknowledge and the Return from Interrupt cycles. The frequency and phase relationship of the SIO clock must be the same as that of the CPU clock.

12-5.1 The Read Cycle

Figure 12-9 shows the timing relationships during a CPU input instruction when the CPU requests a data or status word from the SIO. Note that the read cycle is initiated by the active status of $\overline{IORQ}$ and $\overline{RD}$ and the inactive status of $\overline{M1}$. The wait state W1 is automatically inserted.

12-5.2 The Write Cycle

Figure 12-10 shows the timing relationships during a CPU output instruction when the CPU writes a data word or a control byte in the SIO chip. Notice that the write cycle is initiated by the active status of IORQ and the inactive status of $\overline{RD}$ and $\overline{M1}$. The wait state is automatically inserted.

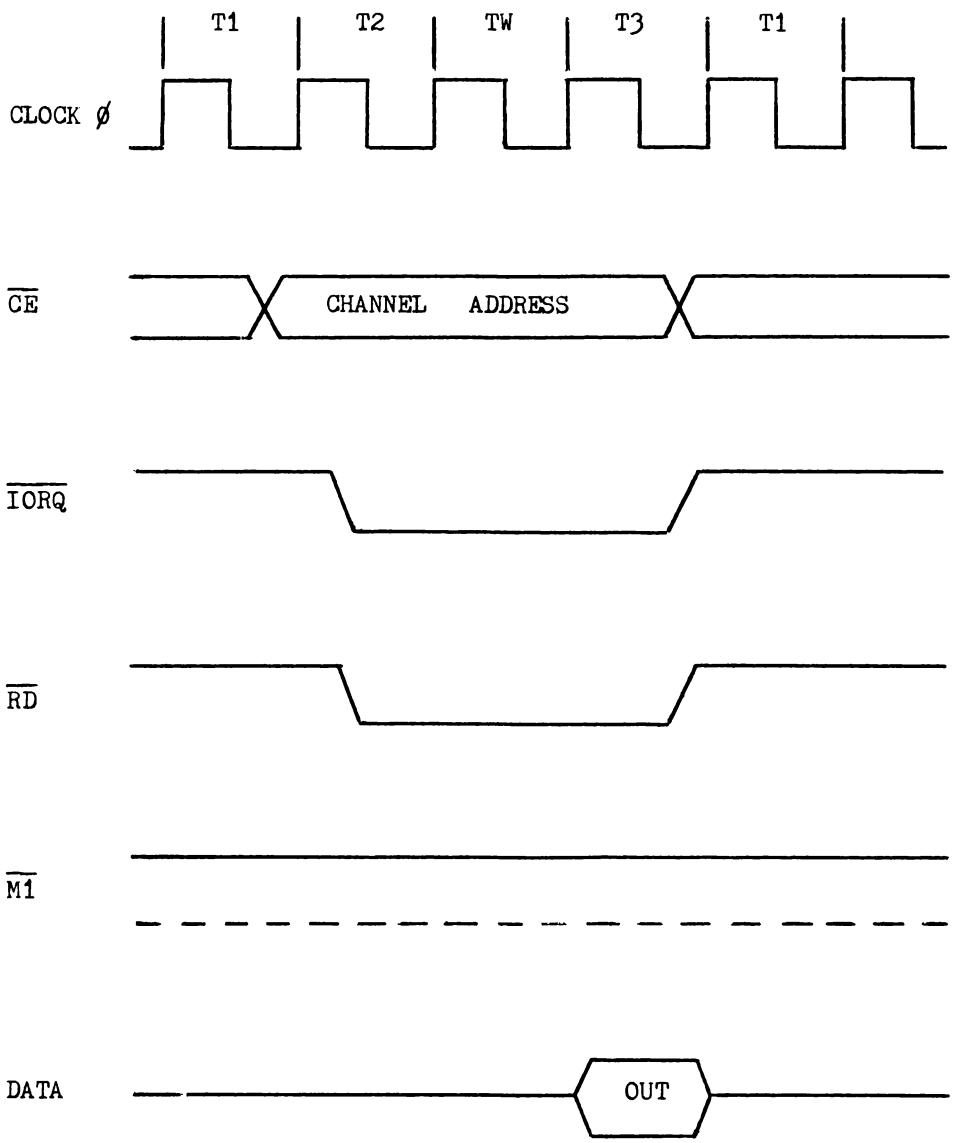


Fig. 12-9 SIO Read Cycle timing.

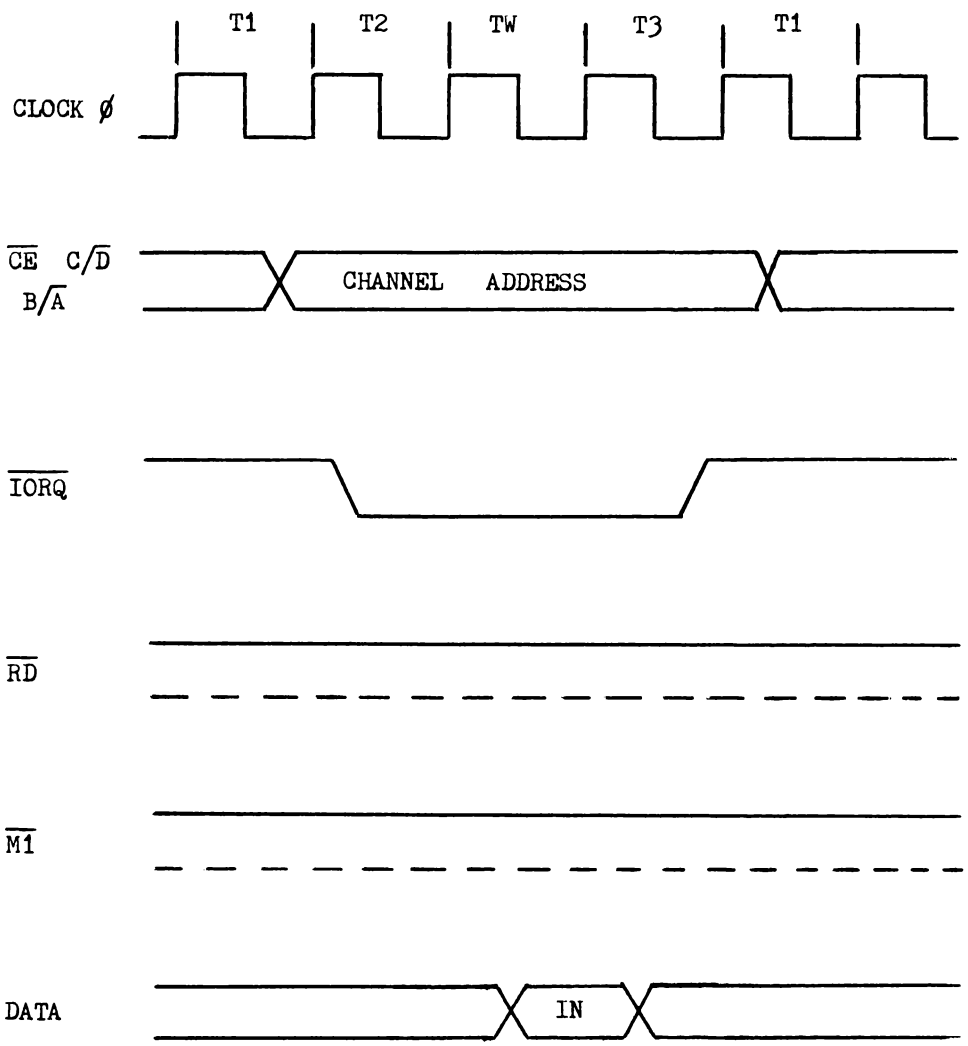


Fig. 12-10 SIO Write Cycle timing.

12-5.3 The Interrupt Acknowledge Cycle

1. An interrupt request to the CPU is indicated by the $\overline{INT}$ line going low.
2. Sometime thereafter the CPU acknowledges the interrupt by the $\overline{M1}$ line going low and a few clock periods later, the $\overline{IORQ}$ going low.
3. The circuits in the daisy chain then determine the interrupt priority rating of the requesting device. The IEI line of the highest priority device is high.
4. The IEO of any device whose interrupt request is being serviced, or is pending, is in the low state.
5. The IEO of devices with no pending interrupts or not currently being serviced, the IEO = IEI.
6. All interrupt signals are prevented from changing while $\overline{M1}$ is low. This ensures stability in the daisy chain.
7. When IORQ is active, the highest priority device requesting interrupt service places its interrupt vector on the data bus and latches in its interrupt request.
8. The timing relationships are shown in Fig. 12-11.

12-5.4 The Return from Interrupt Cycle

1. A Return from Interrupt Instruction (RETI) is inserted at the end of every interrupt service subroutine.
2. RETI is a 2-byte OP code (ED-4D). It resets the interrupt-under-service flip-flop thereby ending the interrupt that has just been processed.
3. Then the first byte of the RETI (i.e. ED) is decoded and the daisy chain forces the IEO high of any interrupt that has not been acknowledged. Thus, the daisy chain asserts that the device that is currently being serviced is the only one with a high IEI and a low IEO.
4. The timing relationships for this cycle are shown in Fig. 12-12.

12-6 NOTES ON CRC OPERATION

12-6.1 Introduction

A brief explanation of how the CRC logic operates, in both the transmitter and the receiver equipment, is given in this section without resorting to any mathematical analysis.

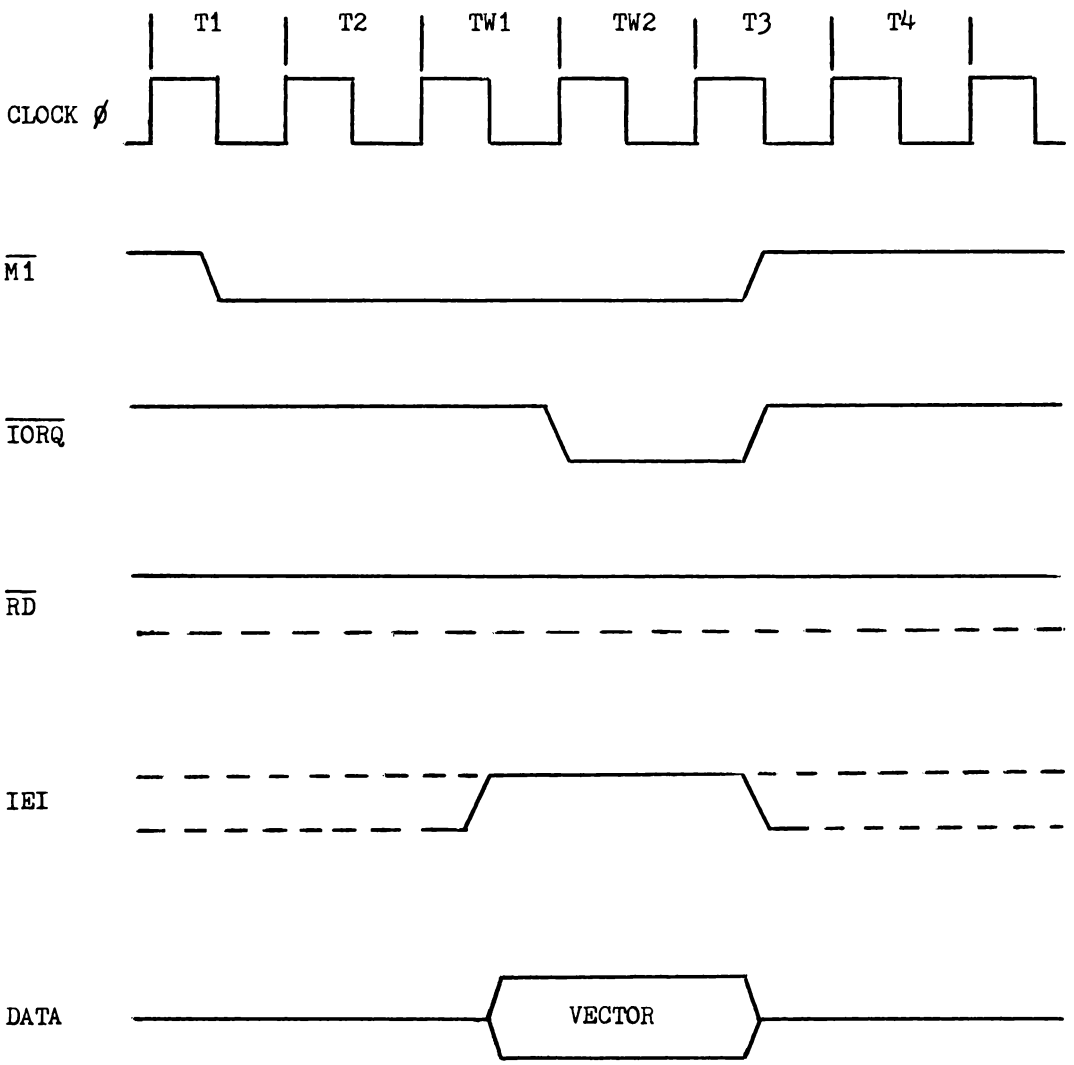


Fig. 12-11 SIO Interrupt Acknowledge cycle timing.

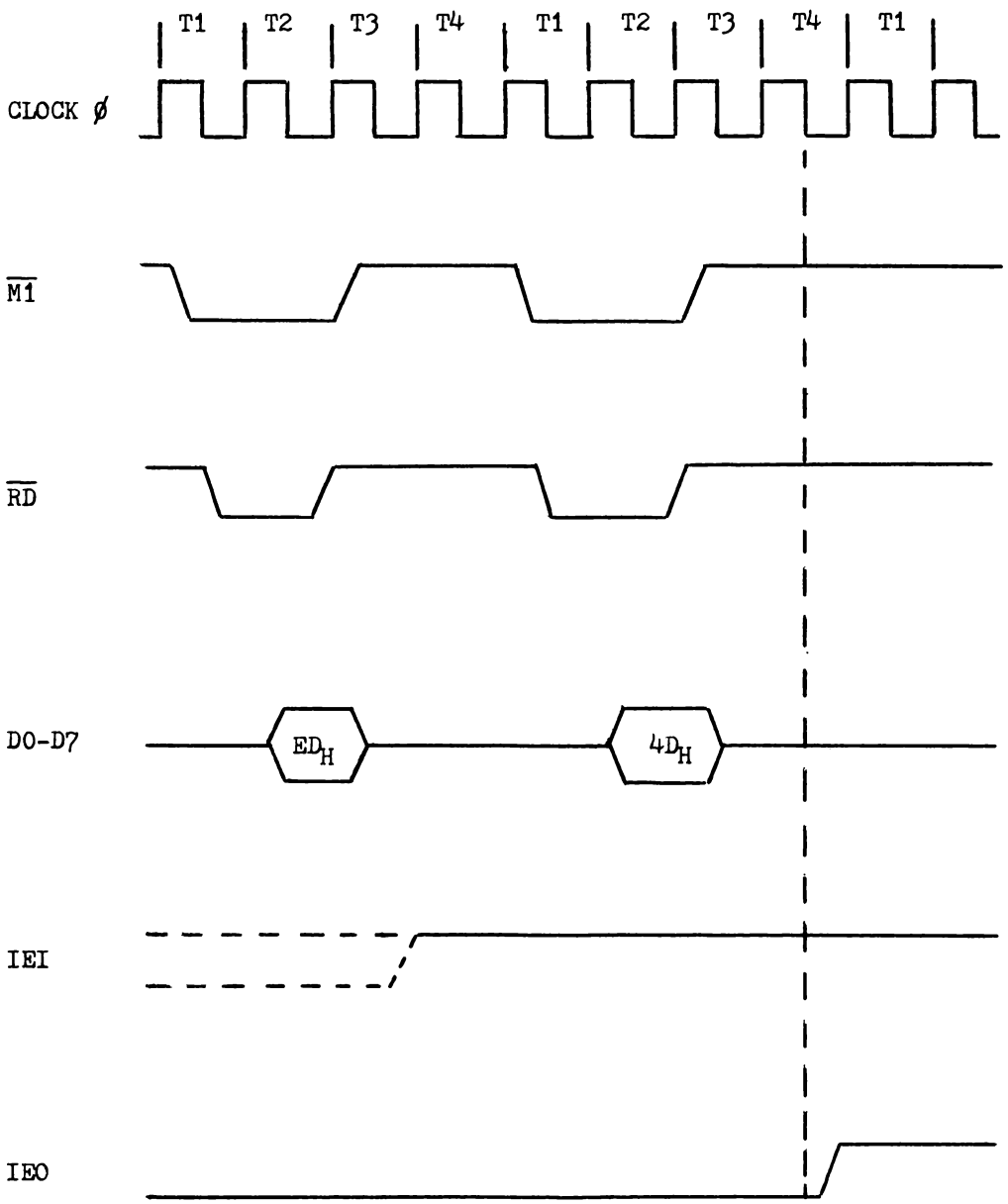
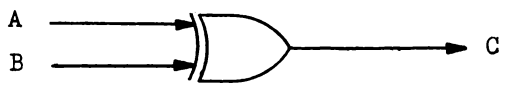


Fig. 12-12 SIO Return from Interrupt Cycle timing.

The CRC operation is based upon a serial shift register and Exclusive-OR circuits. Students will recall that an Exclusive-OR circuit, such as the one shown in Fig. 12.13a, has outputs as shown in the truth table of Fig. 12-13b, for various combinations of input signals.



(a)

I N P U T S		OUTPUT	RULES
A	B	C	No.
0	0	0	1
1	0	1	2
0	1	1	2
1	1	0	1

(b)

Fig. 12-13 The Exclusive-OR. (a) Circuit Symbol; (b) The truth table.

12-6.2 Principle of Operation

The basic steps involved in the CRC operations are explained below:

1. At the transmitting end, two 8-bit CRC characters are generated and appended to the Data/Record field as shown in Fig. 12-4.

2. The compositions of this 16-bit CRC code is determined by the combination of 1s and 0s in the Data/Record field which is to be serially transmitted.

3. The receiver also has a CRC code generator, the same as in the transmitter.

4. As the serial bits of the Data/Record field are transmitted out over the long line, they are also simultaneously fed into the CRC generator, one bit at a time. The CRC code then follows the Data/Record field as shown in Fig. 12-4.

5. At the receiving end, the bits in the Data/Record field are handled as explained in Section 12-3.2 (See Fig. 12-2) and another CRC code (16 bits) is generated as the incoming bits are received.

6. When the CRC code, that is generated by the transmitter, is received, it is compared with the CRC code that is generated by the receiver.

7. If there is an error involved in the transmission, then the CRC codes (one generated by the transmitter and the other by the receiver) will not be identical and an error will be indicated. A match between the two CRC codes will indicate an error-free transmission. This process is shown in a generalized block diagram form in Fig. 12-14. The CRC code generator is explained in Section 12-6.3.

12-6.3 The CRC Generator in the Transmitter

1. The CRC generator consists of a multi-sectional, serial shift register whose output is fed into an Exclusive-OR gate. This is X01 gate in Fig. 12-15.

2. Additional exclusive-OR gates are inserted between the various section of this shift register. These are the X02 and X03 gates shown in Fig. 12-15.

3. The output from X01 is fed back into one of the inputs each of X02 and X03 and into the MSB position of the third segment of the serial shift register, i.e. bit position 15.

4. The number of exclusive-OR gates and their placements in the segmented shift register varies, depending upon the code used. This is determined mathematically.

5. Before transmission of the data begins, the serial shift register is cleared to zeros.

6. The first serial bit of the data stream is then applied to one of the inputs of X01 at point M. The output of X01 is fed back into X02 and X03 and bit position 15 of the register.

7. Simultaneously, a shift pulse is applied to the shift register. Thus, if the first bit of the data stream were a 1, a digital 1 would be loaded into bit positions 4, 10 and 15 of the register.

8. As subsequent bits of the data stream are fed at point M and shift pulses applied, a 1 or a 0 will be outputted by each of the exclusive-OR gates, depending upon

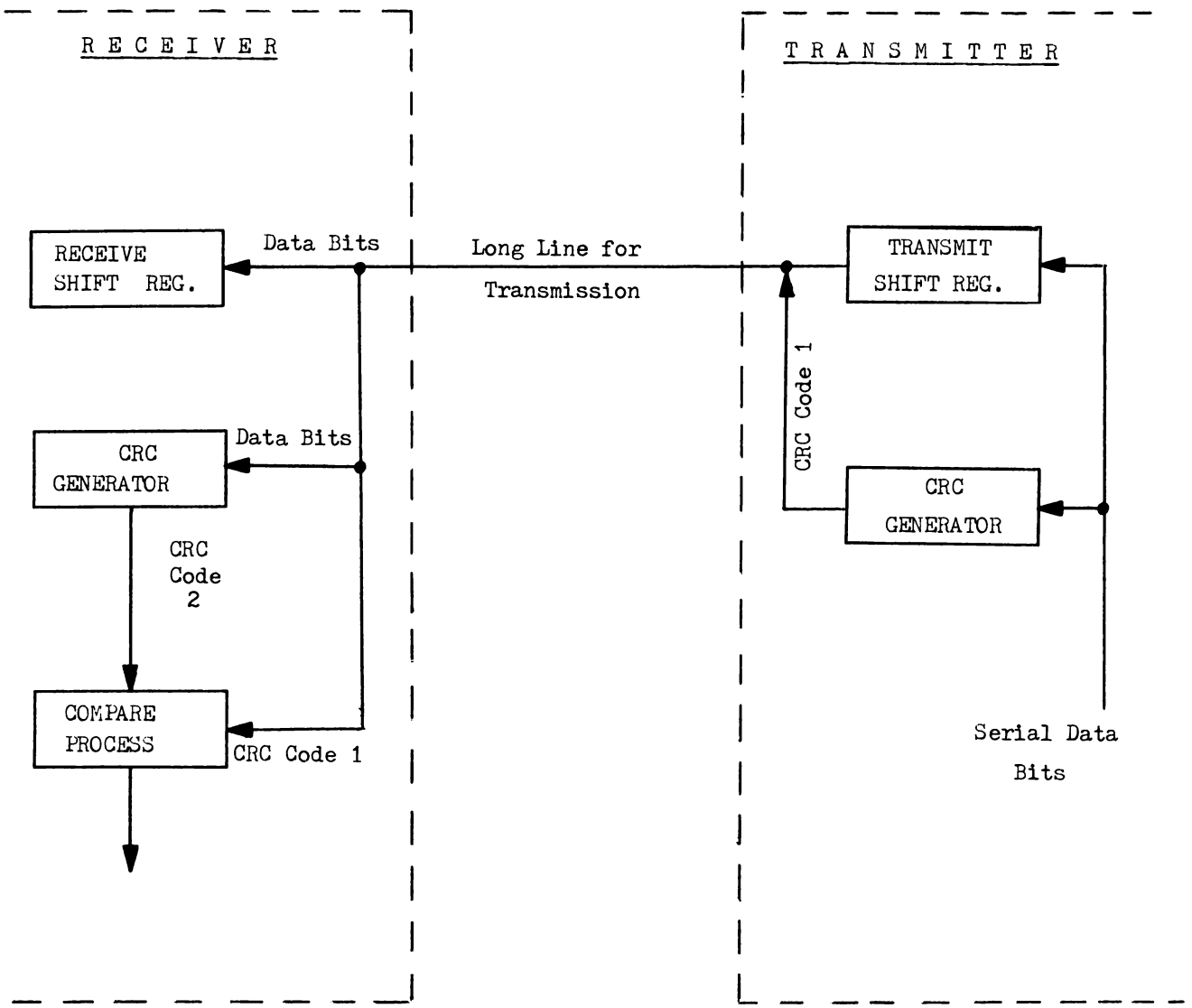


Fig. 12-14 CRC code in transmission and reception.

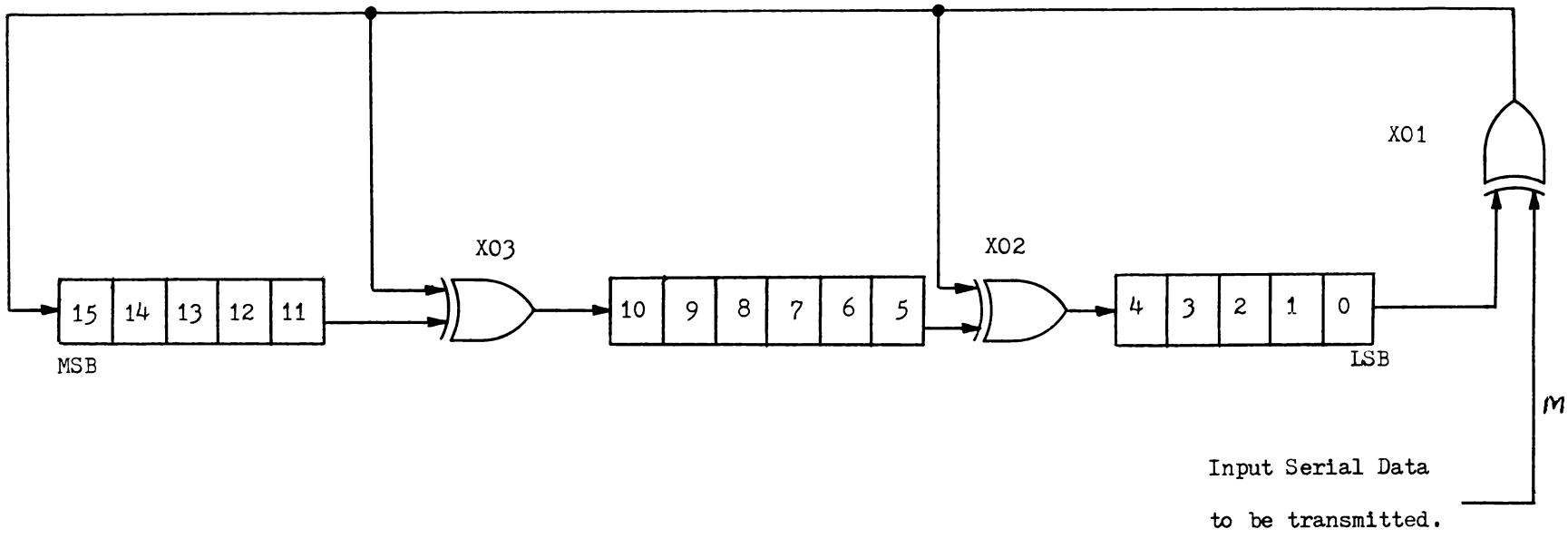


Fig. 12-15 The CRC Generator.

the input at M and the inputs from previous bit positions in the register.

9. When 16 bits are inputted at M, a 16-bit CRC code is generated in the register, depending upon the combination of is and Os inputted at M.

10. After the original data word is transmitted, the contents of the register are serially shifted out and put on the communications line.

12-6.4 The CRC Generator in the Receiver

1. The CRC generator in the receiver is exactly the same as the one in the transmitter.

2. The incoming data bits on the communications line are fed into the Receive Shift Register and simultaneously into the CRC generator.

3. The CRC code is then created by the generator in exactly the same manner as was described in Section 12-6.3 for the transmitter.

4. When the transmitter-generated CRC code arrives at the receiver end, it is compared with the receiver-generated CRC code. If the transmission is error-free, then the two CRC codes (one generated by the transmitter and the other by the receiver) must match exactly, Any discrepancy between the two will indicate an error.

12-6.5 The Comparison Process in the Receiver

1. The actual comparison of the two CRC codes in the receiver can be done by feeding both the CRC codes into a comparator, as shown in Fig. 12-14. The discrepancy between the two CRC codes, if any, is then detected by the comparator.

2. In reality, a more sophisticated method for comparison is used, without the use of a comparator. However, before we describe this method, let us take a look at some of the properties of the Exclusive-OR gate as shown in Fig. 12-16.

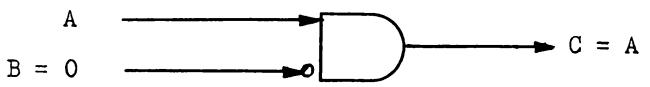
3. The truth table of the Exclusive-OR (Fig. 12-13b) shows two rules which are defined below:

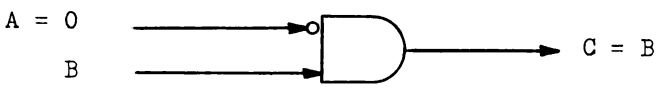
   Rule 1 If both inputs to the Exclusive-OR are the same, then the output is always a 0.

   Rule 2 If one of the inputs to the Exclusive-OR is a 0, the output is the same as the other input.

4. What Rule 1 says is that when both inputs to the Exclusive-OR are the same, this gate essentially acts like a two-input AND gate, with one input being inverted

a. Rule 1.





b. Rule 2.

Fig. 12-16 Rules 1 and 2 for the Exclusive-OR.

and the two inputs tied together as shown in Fig. 12-16a. It is readily seen that the output is always a 0, regardless of the input.

5. What Rule 2 says is that if one of the two inputs is always a 0, then the Exclusive-OR gate again acts like a 2-input AND gate with one inverted input as shown in Fig. 12-16b. The inverted input is always held at 0. The output then reflects the digit 1 input at the other terminal of the AND gate.

6. The implementation of the above two rules enables us to perform a comparison of the two CRC codes without the use of a comparator.

7. This is accomplished by feeding the incoming, transmitter-generated, CRC code directly into the CRC generator of the receiver.

8. When the first bit of the transmitter-generated CRC code is applied at point M of the generator (Fig. 12-15), gate X01 follows the above-mentioned Rule 1 and outputs a 0 provided the first bit of the two CRC codes match.

9. The rest of the Exclusive-OR gates in the generator follow Rule 2. Note that the 0 output of X01 is now shifted into the MSB position of the register.

10. Subsequent incoming bits at point M follow the same procedure and the output of X01 continues to be a 0 as long as the corresponding bits of the two CRC codes match. Thus after all 16 bits are compared, the shift register contains all 0s, thereby indicating a perfect match between the two CRC codes.

12-7 REVIEW QUESTIONS

12-1 Indicate true or false for the SIO chip.

a. ___ The SIO contains 2 fully independent, simplex, communications channels.

b. ___ The SIO interfaces the Z-80/CPU with Modems and other I/O devices which require parallel digital communications capabilities.

c. ___ The SIO functions as a serial-to-parallel and parallel-to-serial converter.

d. ___ The SIO handles both synchronous and asynchronous protocols.

e. ___ The SIO handles only the byte-oriented protocols.

12-2 In the SIO, receiver data are _____ buffered and the transmitter data are _____ buffered. (Single; double; triple; quadruple)

12-3 Indicate the data rates at which the SIO can operate in the synchronous mode.

a. _____ With 2.5 MHz system clock.

b. _____ With 4.0 MHz system clock.

(0 - 880; 150 - 880; 0 - 150; 0 - 550; 50 - 550 Kbits per second)

12-4 Circle below the number of bits per character that the SIO can handle.

a. 6
b. 10
c. 12
d. 8
e. 16
f. 5

12-5 Answer yes or no for the capabilities of the SIO chip.

a. ___ Can it work with 2 stop bits?

b. ___ Is it possible to detect framing and overrun errors?

c. ___ Can it operate at x8 the clock rate?

d. ___ Can it operate at X64 the clock rate?

e. ___ Is it possible to have more than three lead bonding configurations?

12-6 Indicate true or false for the SIO package pin functions.

a. ___ The D0 - D7 bus is used for data and command transfers between the CPU and the I/O devices.

b. ___ During the write cycle, the SIO accepts data or command words from the CPU.

c. ___ When $\overline{\text{RESET}}$ is active, it enables transmitter and receiver.

d. ___ When $\overline{\text{RESET}}$ is active, it disables all interrupts.

12-7 Circle below the statements that are applicable to the $\overline{\text{IORQ}}$ signal.

a. Initiates the write cycle when $\overline{\text{CE}}$ is inactive.
b. Initiates the write cycle when $\overline{\text{CE}}$ is active.
c. Initiates the read cycle when $\overline{\text{CE}}$ and $\overline{\text{RD}}$ are active.
d. During the write cycle, $\overline{\text{IORQ}}$ and $\overline{\text{RD}}$ are active while $\overline{\text{CE}}$ is inactive.

12-8 Which of the following signals is pulled low by the SIO chip when it is requesting an interrupt?

a. $\overline{\text{IEI}}$
b. $\overline{\text{INT}}$
c. $\overline{\text{IEO}}$
d. $\overline{\text{IORQ}}$

12-9 The following statements relate to the IEI and the IEO signals. Indicate true or false.

a. ___ IEI is used in the system-wide priority interrupt daisy chain.

b. ___ When IEI is high, it indicates that a device with higher priority is currently being serviced.

c. ___ IEO is inactive when IEI is active.

d. ___ When IEO is inactive, it blocks interrupt signals from lower priority I/O devices.

12-10 Which of the following package pins will be active during transmission of serial data via Channel B?

a. T x DA
b. R x CB
c. T x DB
d. R x CA

12-11 The incoming data into the SIO is sampled on the _____ going edge of the _____ signal. (Positive; negative; T x D; R x C; R x D)

12-12 Which of the following signals are buffered by Schmitt Triggers?

a. R x DA and R x DB
b. R x CA and R x CB
c. T x DA and T x DB
d. T x CA and T x CB

12-13 The following statements apply to the SYNCA and SYNCB signals. Indicate true or false.

a. ___ In the synchronous mode, transition of this signal affects the SYNC/HUNT status bits in the RR0.

b. ___ After the sync pattern is detected, the SYNC input must be activated after waiting for 2 full R x C cycles.

c. ___ In the Bisync mode, these pins are active as outputs during the portion of the R x C cycle in which sync characters are recognized.

12-14 The following statements apply to the RTSA, RTSB, CTSA and CTSB signals. Indicate true or false.

a. ___ When used in the asynchronous mode, RTSA and RTSB follow the state of the RTS bit.

b. ____When the RTS bit is set, RTSA and RTSB go low.

c. ___ Low signal on CTSA or CTSB enables the corresponding transmitter.

d. ___ The SIO sends an interrupt request to the CPU when the levels of either CTSA or CTSB change.

12-15 _____ Is it possible to use the DTRA pin as a general purpose output pin. (Yes or no)

12-16 Indicate those lines that are buffered by Schmitt Triggers.

a. $\overline{DCDA}$
b. $\overline{DCDB}$
c. $\overline{DTRA}$
d. $\overline{DTRB}$

12-17 Refer to the Generalized SIO Block Diagram of Fig. 12-1 and indicate true or false.

a. ___ Modem control signals are general-purpose and can be used for functions other than controlling modems.

b. ___ The Interrupt Control Logic provides automatic interrupt vectoring capabilities.

c. ___ Channels A and B have equal priorities in the SIO.

d. ___ The External Status has higher priority than the Transmit Status.

e. ___ The transmit status has higher priority than the Receive Status.

f. ___ The Receive Status has the highest priority.

12-18 Refer to the Read Cycle Timing in Fig. 12-9, The Read cycle is initiated by:

a. The inactive status of $\overline{M1}$ and the $\overline{RD}$ and the active status of $\overline{IORQ}$.
b. The active states of $\overline{M1}$ and the inactive states of $\overline{RD}$ and $\overline{IORQ}$.
c. The inactive status of $\overline{IORQ}$ and the active status of $\overline{CE}$ and $\overline{M1}$.
d. The inactive status of $\overline{M1}$ and the active status of $\overline{IORQ}$ and $\overline{RD}$.

12-19 Refer to Fig. 12-11 and indicate true or false for the SIO Interrupt Acknowledge cycle.

a. ___ The IEO of any device whose interrupt request is being serviced, or pending, is in the high state.

b. ___ All interrupt signals are prevented from changing while M1 is low.

c. ___ When $\overline{IORQ}$ is low, the highest priority device. requesting interrupt service, places its interrupt vector on the data bus.

12-20 Refer to the Receive Data Path block diagram of Fig. 12-2. If the incoming character is 5 or 6 bits long and is transmitted in the asynchronous mode, then,

a. It enters the 3-bit buffer and is then loaded into the 8-bit Receive Shift Register.

b. It enters the 3-bit buffer, moves into the SYNC Register Zero delete and is then loaded into the Receive Shift Register.

12-21 Refer to Fig. 12-2 and indicate true or false for the asynchronous transmission mode.

a. ___ If the incoming word is 7 bits long plus parity, the parity bit is stripped.

b. ___ The receive logic automatically inserts Os when a character of less than 8 bits is received.

c. ___ If the incoming character is 7 bits long, it bypasses the 3-bit buffer.

d. ___ The output of the Receive Shift Register is always an 8-bit parallel word.

12-22 From the statements below pick out those that apply to the synchronous receive mode.

a. The sync pattern is stored only in WR6 for the Bisync synchronization of character.

b. The data character is captured and synchronized only when a match is achieved between the 2 SYNC patterns and the contents of WR6 and WR7.

c. The SIO first goes through a HUNT phase regardless of whether the Monosync or the Bisync mode is used for character synchronization.

12-23 Indicate true or false for the CRC logic shift register.

a. ___ It is a serial-input/parallel-output shift register.

b. ___ It is a parallel-input/parallel-output shift register.

c. ___ It is a serial-input/serial-output shift register.

d. ___ It is a parallel-input/serial-output shift register.

12-24 Fill in the blanks for the CRC codes.

a. In error-free serial transmissions _____ CRC codes are generated in the receiver and the transmitter. (Same; different).

b. When a match between the two codes is obtained, the receiver serial shift register will contain _____. (transmitter-generated CRC code; receiver-generated CRC code; all zeros)

12-25 Refer to the CRC generator of Fig. 12-15 and indicate true or false.

a. ___ The register may contain more than 16 bit positions.

b. ___ The MSB segment and the LSB segment must contain the same number of bit positions.

c. ___ It is possible to insert more than two Exclusive-OR gates in the register.

d. ___ The completed CRC code is always shifted out of the register (for serial transmission) from the LSB end.

ANSWERS TO SELECTED REVIEW QUESTIONS

Chapter 1 THE Z-80 CPU HARDWARE ARCHITECTURE

1-3 a. True
b. False
c. True
d. True

1-5 Daisy Chain Logic

1-8 8; 8

1-11 a. False
b. False
c. False
d. True

1-15 d. Register R

1-18 c. When the MSB of the corresponding accumulator is a 1.

1-21 a. False
b. True

1-24 Register Indirect Addressing Mode.

1-27 a. False
b. True
c. False
d. True
e. True

1-30 a. False
b. True

Chapter 2 THE Z-80 INSTRUCTION SET

2-2 Load Immediate

2-5 Accumulator

2-8 a. True c. False
b. False d. False

2-11 d. LD r_1, r_2

2-14 dd

2-17 a. (SP - 1) ← IY_H

(SP - 2) ← IY_L

2-20 A - (HL)

HL ← HL - 1

BC ← BC - 1

Chapter 3 THE Z-80 INSTRUCTION SET

3-1 a. False c. False
b. True d. True

3-4 a. LSB; MSB b. LSB

3-7 RLC (IX + d); 4

3-10 a. False c. True
b. True d. False

3-13 True; PC; incremented.

3-16 1

3-19 a. False c. False
b. False d. True

Chapter 4 The Z-80 INSTRUCTION SET & INTERRUPT RESPONSE

4-2 a. True c. False
b. False d. True

4-5 B; HL

4-8 a. True c. True
b. False d. True

4-11 a. B = 0 c. Decremented
b. B ≠ 0; B = 0

4-14 a. CPL

4-17 DI; resets

4-20 b. INT

Chapter 5 THE Z-80/CPU TIMING AND CONTROL

5-1 b. The T cycle

5-4 a. No Yes
b. Yes

5-7 c. $\overline{\text{MREQ}}$

5-10 a. False c. True
b. False d. True

5-13 a. True c. True e. True
b. False d. True

5-16 c. It assures sufficient time for the memory to write into the addresses location.

5-19 a. False c. False
b. True d. True
e. True

5-22 $\overline{\text{IORQ}}$

5-25 Leading; last; last.

Chapter 6 Z-80 PARALLEL INPUT-OUTPUT CHIP

6-2 a. True c. False
b. True d. True

6-5 Bit Control Mode (Mode 3)

6-8 D0 - D7

6-11 a. Data; A b. Positive; trailing.

6-14 a. Input; $\overline{\text{STROBE}}$ c. $\overline{\text{IORQ}}$
b. $\overline{\text{INTA}}$

6-17 a. True c. True
b. True d. False
e. True

Chapter 7 PROGRAMMING THE PIO CHIP

7-1 a. True c. True
b. False

7-4 a. 8 8 d. E E
b. D 0 f. A 6

7-7

Bit positions	D7	D6	D5	D4	D3	D2	D1	D0
Byte Code	1	0	0	1	0	1	0	0

7-10 c. The assignment of the status and control lines of Port A is arbitrary.

7-13 a. False c. False
b. False d. False

7-16 a. Mode 2 requires only the Mode Set byte and the Interrupt Vector.

Chapter 8 Z-80 COUNTER/TIMER CIRCUIT CHIP

8-2 a. Yes c. Yes
b. Yes

8-5 a. False c. True
b. True d. False
e. True

8-8 a. False c. True
b. True d. True
e. False

8-11 $\overline{CE}$; $\overline{IORQ}$; $\overline{RD}$

8-14 $\overline{RESET}$

8-17 a. True c. False
b. False d. False

8-20 a. Product c. Prescaler; Down Counter
b. Counters; Divides

Chapter 9 PROGRAMMING THE CTC CHIP

9-1 b. Mode 2

9-4 a. 0

9-7 b. D0 and D1

9-10 b. $3D04_H$

9-13 c. $\overline{INT}$

9-16 d. $3C_H$ and 35_H

9-18 c. 921.6 µs

Chapter 10 Z-80 DIRECT MEMORY ACCESS CHIP

10-2 b. The starting address can remain fixed, if so desired.
d. The starting address can be automatically incremented.

10-5 a. Yes Yes
b. No

10-8

a. on command (Load)
c. Automatically (Auto Restart)

10-11 a. False c. True
b. False d. True

10-14 a. $\overline{INT}$ c. IEO
b. IEI

10-17
a. False
b. True
c. False
d. True
e. False
f. True

10-20
a. Initialization
b. Match word

10-23
b. $\overline{\text{CE/WAIT}}$

10-26
a. False
b. False
c. True
d. True

10-29
a. Yes
b. Yes
c. No

Chapter 11 PROGRAMMING THE DMA CHIP

11-1
a. 7; 14
b. Base
c. Associated

11-4
a. True
b. False
c. False
d. False
e. True
f. False

11-7

WR3-0	1 1 0 1 1 1 0 0
WR3-1	1 0 0 0 0 0 0 1
WR3-2	1 1 1 1 1 1 1 1

11-10 1 1/0 1 1 0 1/0 1 0

11-13
a. False
b. True
c. False
d. True

Chapter 12 Z-80 SERIAL INPUT OUTPUT CHIP

12-2 Quadruple; Double

12-5
a. Yes
b. Yes
c. No
d. Yes

12-8 $\overline{INT}$

12-11 Positive; $\overline{R \times C}$

12-14
a. False
b. True
c. True
d. True

12-17
a. True
b. True
c. False
d. False

12-20 a. It enters the 3-bit buffer and is then loaded into the 8-bit Receive Shift Register.

12-23
a. False
b. False
c. True
d. False

INDEX

A

ADD 9
Addressing mode
  bit 13
  extended (direct) 14
  immediate 14, 32
  immediate extended 14
  implied 13, 32
  indexed 8, 14
  indirect 9, 206
  modified page zero 15
  register 13
  register indirect 13
  relative 14
Algorithm 12
Arithmetic Logic Unit (ALU) 2, 9
Arrays 4
Assembler 3
Auto Restart 239

B

Binary-coded-decimal (BCD) 4, 10
Bit
  Carry 12
  Half Carry 10
  Overflow 12
  Parity 12
  Reset 10
  Sample 10
  Set 10
  Sign 10
  Subtract 12
  Test 10
  Zero 10
Bit control, individual 130
Bit control mode 131
Bit manipulation 4
Block lengths 237
Block transfers of data 4
BRANCH 7
Buffering
  double 268
  quadruple 268
Bus acknowledge 238
Bus control 225
Bus Priority logic 225
Bus request 238
Byte control mode 130
Byte counter 227
Byte input, unidirectional 130
Byte manipulation 4
Byte, set interrupt control 161
Byte transfer application 167
Byte output, unidirectional 130
Bytes, command and control 238

C

Carry flip-flop 31
Character
  Bisync 292
  CRC 293
  Monosync 292
Character string 4
Clock period 106
Codes, register pair 36
Comparator 227
Compiler 3
Control word 131
Converter
  Parallel-to-serial 268
  Serial-to-parallel 268
Counter/Timer Circuit (CTC) 2, 175
Counter, down 180
CRC
  generator 315
  initialization 287
  operation 314
CTC counter mode 194
CTC Read cycle 194
CTC Timer mode 196
CTC Write cycle 192
Cycle
  Bus request/acknowledge 116
  Exit from HALT instruction 122
  Instruction 106
  I/O read 113
  I/O write 113
  M 106
  Maskable interrupt
    request/acknowledge 119
  Memory read 111
  Memory write 111
  Non-maskable interrupt
    request/acknowledge 119
  OP code fetch 107
  Stealing 2
  T 106

D

Daisy chain 2
Daisy chain, extending the 150
Debugging, real-time 3
Decimal Arithmetic Adjust (DAA) 12
Decrement 9
Depletion mode 3
Development system (hardware/software 3
Direct Memory Access (DMA) chip 2
Disk, floppy 3
Disk operating system (DOS) 3
Displacement 14
Displacement byte 8

DMA mode,
Burst 224, 237
Byte-at-a-time 224, 236
Continuous 224, 236
Transparent 237
DMA operation
Search only 223, 236
Search-Transfer combined 223, 236
Transfer only 223
DMA Transfers
I/O-to-I/O 236, 247
I/O-to-memory 236, 243
Memory-to-I/O 236, 242
Memory-to-memory 236, 243
Dual-in-line (DIP) package 4, 15, 134
Dynamic memory chips 4

E

Effective memory address 14
End-of-block 224, 247
Error
Framing 287
Overrun 287
Parity 287
Exclusive-OR 314
Execution time, instruction 4
Executive firmware 3

F

File maintenance 3
FIFO (First-in, first-out) 291
Fixed part of mnemonic 35
Flag
Auxiliary carry 12
End-of-block 227
Ready 143
Full Duplex 268
Functions
bit-oriented 268
byte-oriented 268

H

Handshake capability 130
Hunt phase 292

I

Increment 9
Industrial control application 163
Instruction
HALT 18
restart 15
Instructions
arithmetic group of 31
basic CPU group of 32, 94
bit manipulation group of 31
block transfer group of 30
CALL group of 15, 31
exchange group of 30
input/output group of 31, 89
JUMP group of 31, 77
load group of 30
logic group of 31
RETURN group of 31
rotate group of 31
search group of 30
shift group of 31
INTEL CORPORATION 2, 30
INTEL 8008 µC 2
INTEL 8080A µC 2, 30
INTEL 8085A µC 2
Internal Data Bus 9
Interrupt acknowledge 150
Interrupt acknowledge cycle timing 197
Interrupt control and priority logic 225
Interrupt enable-disable flip-flop 97, 162
Interrupt
maskable (INT) 96
multi-level priority 134
non-maskable (NMI) 96
pending 225
request ($\overline{\text{INT}}$) 18
servicing 150
system 96
Interrupts, multiple-level 8

J

JUMP 7

L

Last-in, first-out (LIFO) 8
Logic
internal control 178, 287
interrupt control 178, 187
Logical functions 9
Look-up tables 4

M

Machine cycle one ($\overline{\text{MI}}$) 17
Match found 224
Mnemonic convention 68
Mnemonic structure 34
Mode
asynchronous 289, 296
bidirectional 145
bit control 134, 148, 161
counter 176, 190
input byte 145
synchronous 292, 296
timer 176, 191
output byte 143
Mode control register 131
Mode set word 160
Modem 268
MOS RAM chips, dynamic 8
MOS RAM chips, static 8
MOSTEK, INC. 2

N

N-channel 3
Nesting 8, 134
Non-maskable interrupt ($\overline{NMI}$) 18
Not ready signal 247

O

OP code 14, 32
OP code construction 70
Operand, implied 35
On Match 250

P

Package pin assignments 15
Page, base 15
Page, zero 15
Parallel input/output (PIO) chip 2, 130
Pattern generation, test 3
Pointer 13, 134
Pointer, address 206
POP operation 6
Port addressing 239
Prescaler 180
Program counter 7
Program status word (PSW) 10
Protocol
    bit-oriented 294
    interrupt control 134, 181
    SDLC (IBM) 294
    synchronous 268
Pulse generation logic 225, 238
Pulse interval 225
PUSH operation 6

R

Refresh address 17
Register
    alternate set 5
    block length 227
    channel control 178
    general-purpose (GP) 4, 5
    index 4
    instruction 10
    interrupt page address 9
    interrupt vector 9, 225
    main set 5
    mask 134, 227
    mask control 134
    match work 227
    memory refresh 8
    special-purpose (SP) 4, 5, 6
    status 225, 272, 287
    sync-character 287
    time constant 180, 205
Return from interrupt timing 199

S

SDLC message format 297
Serial input-output (SIO) chip 2, 268
Shifts
    arithmetic 31, 66
    logical 31, 67
Silicon-gate 3
Simulator 3
Software
    diagnostics 3
    support 3
Stack Pointer 7
Static memory chips 4
STROBE signal 143
Subtract 9
Subroutine, service 9
Symbolic expressions 35
Symbols
    register/memory (for rotate/shift) 68
    status register 36
Synchronization, external 292

T

Table look-up 8
Text editor 3
Timing
    bus release (burst/continuous mode, DMA) 247
    bus release (byte-at-a-time mode) 250
    bus request and acceptance (DMA) 247
    DMA active state 242
    DMA inactive state 240
    variable cycle 240
Throughput rate 4
Trigger pulse 196
Tristate busses 16
True memory address 14
Twos complement 8, 14

U

UART 268
USART 268

V

Variable cycle length time 239
Variable part of mnemonic 35
Vector, interrupt 134, 161, 178, 182, 190, 206

W

WAIT states 107
Word, channel control 180, 190, 205
Words, command for DMA set-up 160
Word, time constant 190, 205, 209

Z

ZILOG, INC. 2

ISBN 0-471-86167-7